

AD-A072 966

WRIGHT STATE UNIV DAYTON OH DEPT OF COMPUTER SCIENCE

F/G 9/2

PDP-11 COMPUTER MONITORING SYSTEM.(U)

OCT 78 J E BRANDEBERRY, L A CRUM, R D DIXON

F33615-76-C-1258

UNCLASSIFIED

AFAL-TR-78-192

NL

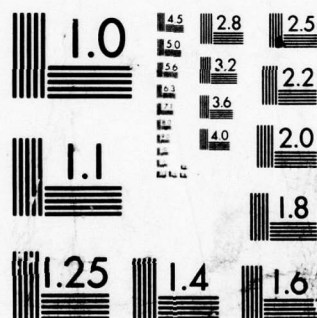
1 OF 1
AD
A072966



END
DATE
FILMED

9-79

DDC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

A072966

LEVEL II

2

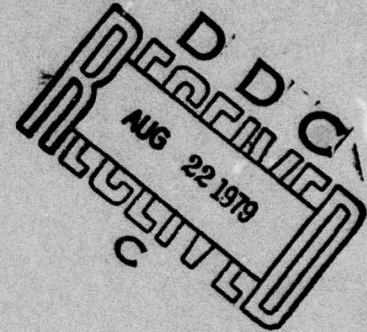


AFAL-TR-78-192

PDP-11 COMPUTER MONITORING SYSTEM

Wright State University
Dayton, OH 45435

J. E. Brandeberry, Ph.D
L. A. Crum, Ph.D
R. D. Dixon, Ph.D
C. B. Ross, Ph.D



October 1978

Technical Report AFAL-TR-78-192

Final Report for period 4/1/76 - 11/30/78

DDC FILE COPY

Approved for public release; distribution
unlimited.

AIR FORCE AVIONICS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433

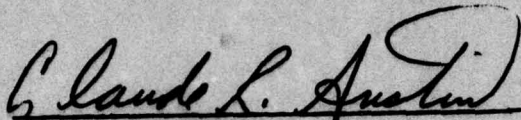
79 08 20 117


NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

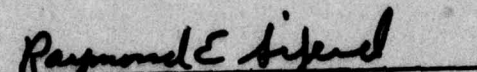
This report has been reviewed by the Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.


CLAUDE R. AUSTIN
Project Engineer


JOHN G. WEBER, Major, USAF
Chief, System Simulation Branch

FOR THE COMMANDER


RAYMOND E. SIFERD, Lt Col, USAF
Chief, System Avionics Division
Air Force Avionics Laboratory

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify AFAL/AAF, W-PAFB, OH 45433 to help us maintain a current mailing list".

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19 REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM	
1. REPORT NUMBER AFAL-TR-78-192	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle) PDP-11 Computer Monitoring System Technical Report		5. TYPE OF REPORT & PERIOD COVERED Final 4/1/76 - 11/30/78	
7. AUTHOR(s) J. E. Brandeberry, L. A. Crum, R. D. Dixon, C. B. Ross		6. PERFORMING ORG. REPORT NUMBER	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Wright State University Department of Computer Science Dayton, OH 45435		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 2003 03 14	
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Avionics Laboratory AFAL/AAF, Wright-Patterson AFB, OH 45433		12. REPORT DATE October 17, 1978	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 27 Oct 78		13. NUMBER OF PAGES 78	
16. DISTRIBUTION STATEMENT (of this Report) Approved for Public release, distribution unlimited. 86 P		15. SECURITY CLASS. (of this report)	
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) 9 Final rept. 1 Apr 76-30 Nov 78,			
18. SUPPLEMENTARY NOTES			
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Accounting Monitoring PDP-11 Microprocessor DEC-10			
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report describes a hardware system for controlling and accounting for use of up to 15 PDP-11's with up to 700 users. The system functions much like a software accounting system on a large time-sharing system and is designed for compatability with DEC system-10 accounting packages. The hardware includes an SBC-80/10 microcomputer interface located in each PDP-11. The interface requires one system unit in the PDP-11 box. The interface becomes "bus master" when the PDP-11 is turned on. It remains			

DD FORM 1473 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

411342

bus master" until the PDP-11 user has correctly "logged on" to the system through the console terminal.

The accounting data and user identification are stored on a PDP-11/04 system which communicates with each SBC-80/10 microcomputer via a serial RS232 communications link. The user identification is generated in a DEC system-10 and transferred to the PDP-11/04 over a serial asynchronous communication link. Account data is periodically transferred to the DEC system-10 via the same communications path.

FOREWORD

This research was performed at Wright State University by a number of graduate students and faculty. The authors would like to thank all people who contributed to this project.

The authors also gratefully acknowledge the assistance and suggestions provided by many people in the Air Force Avionics Laboratory, Wright-Patterson Air Force Base, Ohio, particularly Mr. Claude Austin and the contractor personnel from Systems Research Laboratory.

A special acknowledgement is given to Mrs. Hale for typing this document.

Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DDC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or special
A	

Table of Contents

1.0	Introduction	1
1.1	Applicable Documents	2
1.2	General Theory	2
1.3	Logon Procedure	5
1.4	Logoff Procedure	6
2.0	Theory of Operation	7
2.1	Microprocessor Interface Unit Hardware	7
2.1.1.1	General	7
2.1.1.2	Configuration of SBC 80/10 I/O Interfaces	8
2.1.1.3	Initialization	10
2.1.2	UNIBUS Utilization	11
2.1.2.1	Obtaining the UNIBUS	11
2.1.2.2	Using UNIBUS	12
2.1.2.2.1	Microprocessor Input Via the UNIBUS	12
2.1.2.2.2	Microprocessor Output Via the UNIBUS	14
2.1.2.3	Releasing the UNIBUS	15
2.1.2.4	Reacquiring the UNIBUS	16
2.2	Documentation for the SBC 80/10 Software	17
2.3	PDP-11/04 Data Logging Hardware	25
2.4	Documentation for the Program ACCT	26
2.5	Documentation for the Program COM	38
2.6	Communications Between MIU and PDP-11/04	75
3.0	Conclusions and Recommendations	77

LIST OF FIGURES

Figure		Page
1.0	System Block Diagram.	3
2.4.1	List Layout	35
2.4.2	Block Layout	35
2.4.3	User Record	36
2.5.1	System Configuration.	41
2.5.2	Apparent System Configuration	41
2.5.3	COM Process Diagram	45
2.5.4	Process Control Block	46
2.5.5	Queue Header.	47
2.5.6	Semaphore	48
2.5.7	TTIP State Diagram.	52
2.5.8	DZIP State Diagram.	55
2.5.9	RECAFILE.	58
2.5.10	User Record	60
2.5.11	Set Up User File.	61
2.5.12	Receive a Block	62
2.5.13	Mapping for DEC-10 Blocks to PDP-11/04 Records.	63
2.5.14	WRUSR	64
2.5.15	Process Block	68
2.5.16	STRAN	69

LIST OF TABLES

Table		Page
2.1.1	I/O Port Address Assignments and Functions.....	8
2.1.2	Parallel Port Functions and Connections.....	9
2.6	PDP-11 To/From Intel 8080 Message Packets.....	76

1.0 Introduction

The system described in this document was developed by Wright State University to provide PDP-11 user information to Air Force management on PDP-11's installed in the Avionics Laboratory. The system was also intended to prevent unauthorized use of the PDP-11's.

After a user has been given access to a PDP-11, the system is to allow him to use any software or hardware he desires without interference. In the event that a user's log on information cannot be validated because of a malfunction in some part of the system, the default condition is to give the user access to the PDP-11.

After a user completes his use of the PDP-11 system he needs to log off of the system. This terminates his use and requires a new user to log on to the system before the system can be used again. Powering down the PDP-11 system will automatically log out the current user and require the new user to identify himself before he is granted use of the PDP-11 system. If the system is to be left in a power-on-state for the next user it is possible to log off by depressing the start switch four times in rapid succession.

1.1 Applicable Documents

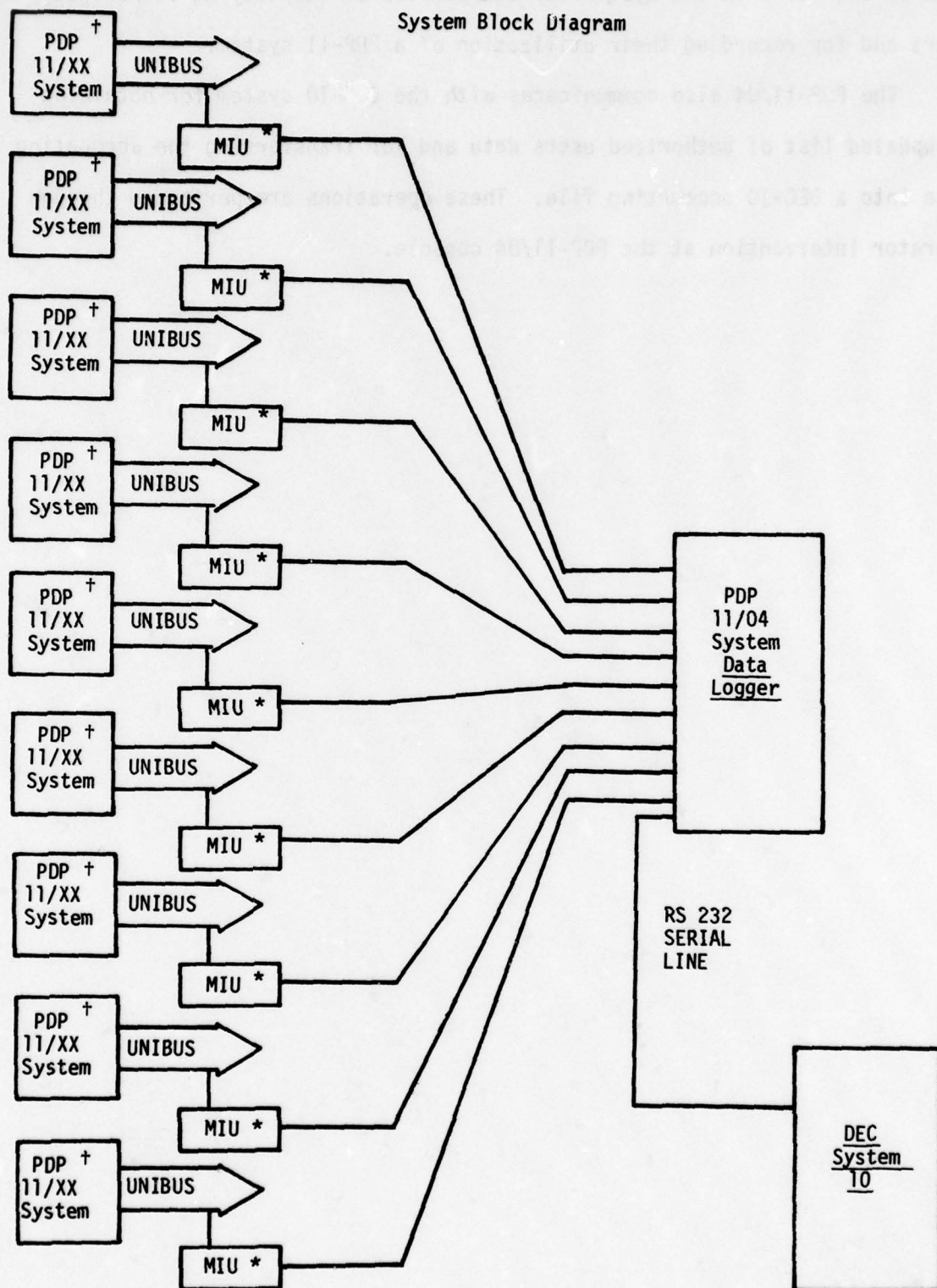
- a. PDP-11 Peripherals Handbook, Digital Equipment Corporation, 1976.
- b. SBC 80/10 and SBC 80/10A Single Board Computer Hardware Reference Manual, Intel, 1977.
- c. Operating and Maintenance Instructions for PDP-11 Computer Monitoring System, Contract No. F33615-76-C-1258, Wright State University, 1978.
- d. PDP-11 Computer Monitoring System Engineering Data for Research and Prototype Hardware, Contract No. F33615-76-C-1258, Wright State University, 1978.
- e. ADM-3 Interactive Display Terminal Operators Handbook, Lear Siegler, 1975.

1.2 General Theory

Figure 1.0 gives a general block diagram of the PDP-11 Monitoring System. It was designed to simultaneously control nine single-user PDP-11 systems, but is easily expandable to fifteen PDP-11 systems.

There are two major hardware components used to implement the monitoring system. One component is the Microprocessor PDP-11 Interface Unit (MIU). An MIU is attached to each PDP-11 which is to be monitored. The MIU is installed in the same manner as any other interface to the PDP-11. The Microprocessor Interface unit contains its own memory and only interfaces to the PDP-11 so that it can inhibit use of the PDP-11 until an authorized user requests the machine. It talks directly to the user over the Unibus through the console device. It does not use the processor, but must be granted a nonprocessor use of the Unibus so that it can take control of the bus and talk to the console device while it is bus master. The information obtained from the user is checked against master files stored on a PDP-11/04 which is the other major hardware component. The function of the PDP-11/04 is to communicate with all

Figure 1.0
System Block Diagram



* Monitoring System Hardware Provided by Wright State University

+ Systems being monitored

nine of the MIU's in the system for the purpose of identifying authorized users and for recording their utilization of a PDP-11 system.

The PDP-11/04 also communicates with the DEC-10 system for obtaining an updated list of authorized users data and for transferring the accounting data into a DEC-10 accounting file. These operations are performed through operator intervention at the PDP-11/04 console.



1.3 Logon Procedure

The PDP-11's installed in the Avionics Laboratory include many different models. All have a Unibus and interface with the same hardware, but due to architectural differences in the various models they may function slightly differently. In some systems the Unibus is inactive during the halt state of the processor. In other systems the Unibus is active and can be used by other devices in a non-processor mode while the processor is halted.

The correct log on procedure is to turn on the PDP-11 processor in the Halt state. Boot the system in the normal procedure for the system. When the processor is started the Microprocessor will take control of the Unibus and require the log on sequence. The program will type the following on the console:

PROJ,PROG-NO. =

A proper response is up to four numbers followed by a comma and four more numbers and Restart. The numbers are echoed on the console, and an example of a correct response is

PROJ,PROG-NO. =1111,11RET

The program will then type:

PASSWORD =

A proper response is from one to nine alphabetical characters followed by Return. The characters in the password are not echoed. If the log on is accepted, control of the Unibus will be returned to the processor and the boot will be executed. When you logged on the system and accounting data was recorded the "Date-Time-LOG ON OK," if the accounting data was recorded. If the accounting

data was not recorded due to some hardware malfunction the microprocessor will print "-LOG ON OK." without time and date. This indicates control was returned to the processor, but accounting data was not recorded.

If the machine is already powered up when you arrive and it has printed on the console:

PROJ,PROG-NO. =

type in your identification and password with the machine in the run mode.

You may need to boot the system after you have log on.

If you have a problem logging on the system and cannot recover, power down the computer and start the initial log on procedure. This resets everything in the PDP-11 and in the log on system.

1.4 Log Off Procedure

Three procedures exist for logging off of the accounting system. One method is to turn off the power to the PDP-11. This causes the PDP-11/04 to log off the current user of PDP-11. In this case, no log off information is printed for the user. The second method is to halt the processor and then depress the Start switch, four times, within a two second interval. Third, any deposit or write operation to M105 address followed by an INIT will cause the log off sequence to be initiated. In these two cases, the user is given the amount of time he used the system and a message indicating that he has been logged off of the system.

On computer systems where the Unibus is not active in the Halt state, it is necessary to load a valid program starting address (possibly the boot) and then starting that program in order to get logged off. The microprocessor interface will only log off a user after the PDP-11 processor has granted the microprocessor a nonprocessor use of the Unibus.

2.0

THEORY OF OPERATION

2.1 Microprocessor Interface Unit Hardware

2.1.1.1 General

The microprocessor accounting system interfaces with the various PDP-11 systems through a specially wired DEC mounting panel, BB-11. On this panel are mounted three standard DEC flip-chips: M7821 (Interrupt Control Module) in slot F1, M105 (Address Selector Module) in slot E1, and M796 (UNIBUS Master Control Module) in slot D1. Slots A1, B1, A4, and B4 are used for connecting the UNIBUS and power is supplied via an external cable, or by means of a paddle card. +5, +15, and -15 volts are required. The board containing the rest of the microprocessor system is mounted in row 3, slots C through F. The Intel SBC 80/10 Single Board Computer, which controls the operation of the interface, is attached to this board and communicates with it by means of a lower edge connector, P1, and an upper edge connector, J1, which is wired to the SBC 80/10 Parallel I/O Interface. Another upper edge connector, J3, is used to connect the SBC 80/10 Serial I/O Interface with the PDP-11/04 Data Logging System.

The sequencing of operations executed by the interface is controlled by the program stored in a 1K x 8 bit Intel 2708 EPROM installed at location A23 on the SBC 80/10 board. A listing of this program is contained in section 2.5 of "Engineering Data for Research and Prototype Hardware," and it is discussed in section 2.2 of this document. In the subsections which immediately follow, the configuration of the SBC 80/10 I/O interfaces, and how the hardware performs the functions of initialization and obtaining - using - releasing - reacquiring the UNIBUS will be discussed.

2.1.1.2 Configuration of SBC 80/10 I/O Interfaces

The operation of the SBC 80/10 Serial and Parallel I/O Interfaces, which consist primarily of an Intel 8251 Programmable Communication Interface (USART) and two Intel 8255 Programmable Peripheral Interface chips respectively, are fully described in the SBC 80/10 Hardware Reference Manual. The serial and parallel port address assignments which are actually used in the present system and their functions are given in Table 2.1.1.

I/O Port Address (hexadecimal)	Interface Type	Function
E4	Parallel	Group 1 - Port A
E5	Parallel	Group 1 - Port B
E7	Parallel	Group 1 - Control
EC	Serial	Data In/Out
ED	Serial	Status In and Mode or Command Out

Table 2.1.1 I/O Port Address Assignments and Functions

The Serial Interface is configured as an EIA RS232C interface. It is used for communication with the PDP-11/04 Data Logging System.

Only 18 of the 48 signal lines of the Parallel Interface are used, all belonging to Group 1 and controlled by the 8255 in location A19. One signal is used for initialization and for releasing the UNIBUS; the others are used for communicating with the user via the UNIBUS. The eight bits of Port A are connected to the eight low-order UNIBUS data lines, via the two Intel 8226 4-Bit Bidirectional Bus Drivers in A1 and A2, and 8226 and 8838 transceivers on the main interface board. Jumper connections 13-14 and 41-43 are made to allow input from Bit 6 of Port C to dynamically change data directions at the SBC 80/10 8226's. Only the low order four bits of Port B are used, always for output. For this purpose, a 7438 chip has been installed in Socket A6. Bit 0

of Port B is the one bit used for initialization. Bit 1 is connected to the C1 CONTROL input of M796. Bits 2 and 3 are directly connected to UNIBUS A01 and A02. Port C is used to transmit control signals between the 8255 and the rest of the interface. The bits are dedicated to different functions depending on the mode of operation of Ports A and B. Only six bits are actually used. Bits 0 and 3 are disabled by breaking jumper connections 44-45 and 50-51. These bits output interrupt requests and are not used because parallel I/O is performed under program control only. This simplification also permits the simultaneous utilization of both Ports A and B in the modes desired. A 7400 chip is installed at A3 and an SBC-902 resistor terminator chip at A4 for driving and receiving the control signals of Port C.

The 8255 functions and interface connections of the parallel port lines in use are summarized in Table 2.1.2.

Port Bit	8255 Function	J1 Pin Number	Interface Connection
A0	Bidirectional Data	43	UNIBUS D00
A1	"	41	" D01
A2	"	45	" D02
A3	"	47	" D03
A4	"	39	" D04
A5	"	37	" D05
A6	"	35	" D06
A7	"	33	" D07
B0	Data Output	7	Internal Reset
B1	"	5	C1 CONTROL (M796)
B2	"	3	UNIBUS A01
B3	"	1	" A02
C1	$\overline{\text{OBF}}(\text{B})$ Output	29	START (M796)
C2	$\overline{\text{ACK}}(\text{B})$ Input	19	ADRS TO BUS (M796)
C4	$\overline{\text{STB}}(\text{A})$ Input	21	DATA STROBE (M796)
C5	$\text{IBF}(\text{A})$ Output	27	ACCEPTED (M796)
C6	$\overline{\text{ACK}}(\text{A})^1$ Input	23	DATA TO BUS (ff 7)
C7	$\overline{\text{OBF}}(\text{A})$ Output	31	MASTER (ff 7)

Table 2.1.2 Parallel Port Functions and Connections

¹Also DATA IN ENABLE of 8226's in A1 and A2.

2.1.1.3 Initialization

The interface uses UNIBUS INIT, which is asserted low whenever the START key on the PDP-11 Console is depressed or the PDP-11 Processor executes a Reset instruction, to request control of the UNIBUS. This ensures that a user cannot run a program unless he is logged on. As this signal is normally used to clear and initialize all peripheral devices on the UNIBUS, the initialization of this interface must be handled by itself.

When power is turned on, the microprocessor immediately begins to run. A RESET signal, generated by the 8224 Clock Generator in A31 from the power supply rise, is used to restore the microprocessor's internal program counter to zero. This pulse, called 8080 INIT on the schematic, is available at Pin 14 of P1. It is used to disable the receiver of UNIBUS INIT for approximately 1msec, locking out any UNIBUS INIT which might occur before initialization is completed. It also presets a flip-flop, chip 7, which permits the passage of one and only one subsequent UNIBUS INIT for obtaining control of the UNIBUS. This flip-flop can also be manually preset from the Console as described below under "Reacquiring the UNIBUS," method (2).

The Serial and Parallel Ports of the SBC 80/10 are then initialized under program control. As turning on the power brings the USART up in Reset status, it expects a mode instruction prior to use. The mode is set to odd parity with two stop bits and eight-bit characters at a baud rate factor of 64. Along with the jumper connection 10-4 on the SBC 80/10 board, this establishes 4800 baud as the serial communication rate. The mode instruction is followed by a command instruction which specifies no hunt mode, forces $\overline{\text{RTS}}$ to zero, and enables both the transmit and receive functions of the USART. The control word 89 (hexadecimal) is then sent to the Group 1 control port of the Parallel Interface, which places the data ports in the following modes:

Port A - Output mode 0; Port B - Output mode 0; Port C - Input mode 0. A word of zeroes is then written out to Port B. Bit 0 provides a low pulse to SACK ENABLE (M7821), which releases the UNIBUS if the interface had obtained it on powering up, and clears the UNIBUS Request flip-flop in M796. This pulse also presets the DATA TO BUS flip-flop which puts the 8226's in A1 and A2 into their input state. A second control word, C4 (hexadecimal) is sent to same control port to reconfigure the data ports for utilization of the UNIBUS as follows: Port A - Bidirectional mode 2; Port B - Output mode 1; Port C - used for control of the other two ports.

After this initialization procedure, the interface is guaranteed not to be holding any UNIBUS lines, but ready to request the UNIBUS upon depression of the START switch. The microprocessor stops execution, and waits for an interrupt.

2.1.2 UNIBUS Utilization

2.1.2.1 Obtaining the UNIBUS (first time)

When the START switch of the PDP-11 console is depressed, the INIT signal generated triggers the M796 Request flip-flop which activates the M7821 Interrupt Control Module. The trigger signal also prevents the interface from requesting the UNIBUS on subsequent INIT assertions, which may occur when the user has been cleared for operation. On PDP-11 systems which are not active in the HALT state, it is necessary to load a valid program starting address before depressing START. Upon completion of the log on procedure, the PDP-11 processor will begin execution of the program. The M7821 generates a non-processor UNIBUS request and handles the arbitration protocol. When UNIBUS mastery is granted to the M7821, it asserts SACK, but holds the UNIBUS until a user has successfully logged on by delaying the unassertion of SACK. The MASTER A signal from the M7821 wakes up the microprocessor and enables START of the M796. The microprocessor is now able to talk with the user via the PDP-11

UNIBUS on the console terminal.

2.1.2.2 Using the UNIBUS

The microprocessor interfaces with the UNIBUS by means of the SBC 80/10 Group 1 parallel ports, using the 8255 and M796 control signals. The connections are listed in Table II above. The only UNIBUS addresses needed are those of the console terminal, 777560-777566 (octal), as given in Appendix A of the PDP-11 Peripherals Handbook. All address bits except A01 and A02 always have the same value and are hard-wired.

In Output mode 1, Port B uses bits 1-2 of Port C for control signals, and Port A in mode 2 uses bits 4-7. The mode definitions of the parallel ports are not changed until the UNIBUS is released. The microprocessor interface needs to read the console status and to input and output data. Typical input and output operations will now be described.

2.1.2.2.1 Microprocessor Input Via the UNIBUS

To perform an input operation, the microprocessor must load the appropriate UNIBUS address and control lines and then read the data lines. It accomplishes this by executing the following sequence of instructions:

```
MVI A, Immediate data for Port B
OUT PORT B
IN PORT A
MOV M, A
```

Valid bit patterns for the data for Port B are: 1011, 0111, or 0011. Bit 0 must be "1" else the UNIBUS will be released. Bit 1 must be "1" since this is an input operation for which C1 CONTROL (M796) must be "0." (The data lines are inverted by the 7438 driver in A6.) The "1"s in Bits 2 and 3 assert the UNIBUS address lines A01 and A02 respectively.

The OUT command sets in motion the following sequence of interface operations. For timing, see the diagram for M796 DATI in the PDP-11 Peripherals

Handbook, and the Mode 1 and Mode 2 diagrams in Chapter 3 of the SBC 80/10 Hardware Reference Manual.

1. When Port B of the Group 1 8255 receives the data from the microprocessor, $\overline{OBF}(B)$ - Output Buffer Full - is asserted low. From Bit 1 of Port C this is gated to produce START in the M796 card which then asserts UNIBUS C0 and C1, ADRS TO BUS, MSYN WAIT, and, after 200ns, MSYN. M796 then waits for the SSYN response from the addressed console device register.
2. ADRS TO BUS (L) is connected to Bit 2 of Port C, $\overline{ACK}(B)$ - Acknowledge. When this line returns to its normal high state, the 8255 negates $\overline{OBF}(B)$. This signal is not used as an external acknowledge to the microprocessor. Since the UNIBUS transactions occur in a fraction of a microprocessor cycle, there is no need for a delay.
3. The assertion of SSYN produces a 200ns pulse that appears as DATA WAIT (M796). The trailing edge of this pulse is used to generate the DATA STROBE (L) pulse which is connected to Bit 4 of Port C, $\overline{STB}(A)$ - Strobed Input. This triggers the 8255 to generate IBF(A) - Input Buffer Full - which is sent via Bit 5 of Port C to ACCEPTED (M796), and causes that card to clear MSYN, thus ending the UNIBUS cycle.

All of the above takes place before the microprocessor executes the IN PORT A command. When this is executed, the microprocessor reads the data last placed in the Port A buffer and removes IBF(A). Bit 6 of Port C, which is connected to $\overline{ACK}(A)$ - Acknowledge - remains high during these operations. This keeps the output buffer of Port A in a high impedance state and the 8226's in A1 and A2 in their input state. No time-out provisions are made if the console device does not respond. The microprocessor proceeds at its own pace to read the 8255

input data buffer, relying on the software and the PDP-11/04 Data Logging System to detect errors or false data. The last command in the sequence given stores the data just read into memory, completing the input.

2.1.2.2.2 Microprocessor Output Via the UNIBUS

For an output operation, the microprocessor first sends out the data and then loads the address and control lines, using the following sequence of instructions:

```
MOV A, M
OUT PORT A
MVI A, Immediate data for Port B
OUT PORT B
```

The first instruction fetches the data from memory. The only valid bit pattern for the Port B immediate data is: 1101. As before, Bits 2 and 3 are address information. The Reset and UNIBUS Release, Bit 0, remains "1." Bit 1 is now C1 CONTROL (M796) for output. The OUT PORT B instruction results in a sequence of events similar to those described under input. There are differences since C1 CONTROL is now "1" at M796, denoting output. The complete sequence of interface operations is as follows.

1. When the microprocessor writes out to Port A, the 8255 control signal, $\overline{\text{OBF}}(\text{A})$ - Output Buffer Full - is asserted low. Called MASTER on the schematic, this signal from Bit 7 of Port C is used to clear the DATA TO BUS flip-flop (chip 7). When DATA TO BUS, connected to Bit 6 of Port C, $\overline{\text{ACK}}(\text{A})$, goes low, the tri-state output buffer of Port A is enabled to send out data. This same signal puts the 8226 Bus Drivers in A1 and A2 into their output state. These load the UNIBUS through two further levels of logic which are controlled by the M796 DATA TO BUS signal.

2. The OUT PORT B command, as before, results in the assertion of $\overline{\text{OBF}}(\text{B})$ which starts the M796 UNIBUS cycle. M796 asserts UNIBUS C0 and C1, ADRS TO

BUS, DATA TO BUS, and MYSN WAIT, followed by MYSN. ADRS TO BUS (L) acknowledges Port B's output, as before, and $\overline{OBF}(B)$ is withdrawn.

3. When SSYN is received from the console device, M796 removes its signals from the UNIBUS. The rising edge of DATA TO BUS (L) clocks the DATA TO BUS flip-flop, which, going high, unasserts $\overline{ACK}(A)$ at the 8255, returning the output buffer of Port A to a high impedance state and the bus drivers at A1 and A2 to their input state.

Again, the microprocessor does not wait for an external acknowledge. Unless the addressed device simply does not answer, the UNIBUS transactions will always be completed in time for the microprocessor. Lack of response is ignored.

2.1.2.3 Releasing the UNIBUS

When a user is to be allowed to use the PDP-11, the interface finally releases the UNIBUS. This is done by returning to the parallel port initialization procedure. As before, control word 89 (hexadecimal) is sent to the control port and zeroes to Port B. This port is now in Output mode 0, but still connected as described. The zeroes in Bits 2 and 3 unassert the UNIBUS address lines A01 and A02. Since the 8255 output buffer is latched, these lines remain unasserted until further data is written out to Port B, which will not occur until the interface again becomes UNIBUS master. Bit 1 goes to the M796 card as before, but is not placed on the UNIBUS as $\overline{OBF}(B)$ is not asserted in mode 0. The "0" in Bit 0 directs M7821 to unassert SACK and clears the UNIBUS Request flip-flop in M796. This releases the UNIBUS. INIT's asserted by the user will not cause the interface to request the UNIBUS except as described in the next section.

Whenever a UNIBUS which is not active in the HALT state is released by this interface, the PDP-11 processor will begin execution in response to

the START command which initiated the interface intervention. At log on and log off time, the user must prepare for this by loading on appropriate starting address (possibly the boot) prior to depressing the START switch.

2.1.2.4 Reacquiring the UNIBUS

In order to log off, the user must return control of the UNIBUS to the microprocessor interface. This can be accomplished in two ways:

(1) By depressing the START switch four times within approximately two seconds. (Actually, any process which asserts UNIBUS INIT four times at least 100 μ sec apart within the two second time period will cause the interface to take over the UNIBUS. If the user anticipates that this condition will occur in normal program operation, this mode of UNIBUS reacquisition should be disabled by changing the +5v applied to pin 2 of chip 18 to ground.)

(2) By depositing any data to the address encoded by jumpers on the M105 card of the interface, followed by depressing the START switch. The deposit enables the passage of the next INIT to the UNIBUS Request flip-flop. When UNIBUS control is reacquired by the microprocessor, it will execute the log off procedure. Then, without releasing the UNIBUS, it will prompt for another log on. The UNIBUS will be released again only at the completion of another log on sequence, or when power is turned off. If a user is logged on when power is turned off, the PDP-11/04 logs off the user, but no log off information is printed.

2.2 Documentation for the SBC 80/10 Software

An algorithmic outline of the program controlling the microprocessor interface follows.

MAIN PROGRAM

Initialize Stack Pointer, USART, interface, and parallel ports

Zero data buffers

Enable interrupts

Halt and wait for start of PDP-11

Repeat until power off

 Log on

 Log off

End repeat

Procedure Log on

Repeat until log on is valid

 Accept data

 Verify data

End repeat

Procedure Accept data

Set PW = not acceptable

Repeat until PW is acceptable

 Set PPN = not acceptable

 Repeat until PPN is acceptable

 Print 'PROJ,PROG-NO. ='

 Wait for user to reply

 Read and echo user's reply (at most 10 characters)

```

I f user typed anything other than numerals, a comma, or
a carriage return, then
  Print ' - TRY AGAIN'
  Zero data buffers
Else if format was incorrect, then
  Print "PROJ,PROG-NO. FORMAT IS PPPP,NNNN"
  Print ' - TRY AGAIN'
  Zero data buffers
Else
  Set PPN = acceptable
  Store Project Number in bytes 1-4 of transmit
    Packet (right justified)
  Store Programmer Number in bytes 5-8 of transmit
    Packet (right justified)
End if
End repeat
Print "PASSWORD"
Wait for user to reply
Read but do not echo user's reply (at most 9 characters)
If user typed a control, then
  Print ' - TRY AGAIN'
  Zero data buffers
Else
  Set PW = acceptable
  Store Password in bytes 9-16 transmit Packet
    (left justified with trailing blanks)
  Fill remainder of transmit Packet (bytes 17-32)
    with blanks
  End if
End repeat
*****

```

Procedure Verify data

Set count = 3

Communicate (with PDP-11/04)

If communication fails, then

Set log on = valid

Print ' - LOG ON OK'

Else (communication succeeds)

If first byte of response Packet is 'V', then

Set log on = valid

Print date and time as received

Print ' - LOG ON OK'

Else (first byte is not 'V')

Set log on = invalid

If second byte of response Packet is 'T', then

Print 'CANNOT LOG ON AT THIS TIME OF DAY'

Else (second byte is not 'T')

If Second byte of response Packet is 'N', then

Print 'INVALID PPN'

Else if second byte of response Packet is 'P', then

Print 'INVALID PASSWORD'

End if

Print ' - TRY AGAIN'

Zero data buffers

End if

End if

End if

Procedure Communicate

Set communication = fails

Repeat until communication succeeds or count is 0

 Transmit

 Receive

 Repeat while ACK is received

 Receive

 End repeat

 If reception is valid, then

 Set communication = succeeds

 Else (reception is invalid)

 Set count = count-1

 End if

End repeat

Procedure Transmit

Send ACK via serial port

Send transmit Packet via serial port

Procedure Receive

Wait for response Packet via serial port

If too much time elapses or parity error is detected, then

 Set reception = invalid

Else

 Set reception = valid

End if

Procedure Log off

Halt and wait for user to initiate log off sequence

Place 'L' into byte 1 of transmit Packet

Set log off = invalid

Set count = 3

Repeat until log off is valid or count is 0

 Communicate (with PDP-11/4)

 If communication fails, then

 Set log off = valid

 Print ' - LOG OFF OK'

 Else (communication succeeds)

 If first byte of response Packet is 'L', then

 Set log off = valid

 Print data and time as received

 Print elapsed time as received

 Print ' - LOG OFF OK'

 Zero data buffers

 Else

 Set count = count-1

 End if

 End if

End repeat

The coding of this program in Intel 8080 Assembly Language can be found in the report "Engineering Data for Research and Prototype Hardware." The program is contained in 1K of programmable read-only memory (chip A23) which is selected for addresses in the range 0000-03FF (hexadecimal). The stack pointer and data buffers are located in 1K of random access memory provided on the SBC 80/10 board. This memory is assigned addresses in the range 3000-3FFF (hexadecimal). Only the upper 82 locations are used for data. The remainder is available to the push-down stack.

When power is turned on, the serial and parallel I/O ports of the SBC 80/10 are initialized. The serial port (USART) is used to communicate with the PDP-11/04 while the parallel ports are used to signal the interface, control the PDP-11 Unibus, and communicate with the user. Since the hardware brings the USART up in Reset status, it expects and requires a mode instruction prior to use for communication. The mode is set to odd parity with two stop bits and eight-bit characters, and a baud rate factor of 64. The mode is followed by a command instruction which specifies no hunt mode, forces (not RTS) to zero, and enables both the transmit and receive function. The three parallel ports, A, B, and C, after being used to place the interface in a state where it does not control the PDP-11 Unibus, are set to operate in the following manner: Port A - mode 2 (Bidirectional), Port B-mode 1 output (Strobed) and Port C - mode 0 output (Basic).

Interrupts are then enabled and the microprocessor halts. It is restarted when the start switch on the PDP-11 is activated. When this occurs, the interface assumes control of the PDP-11 Unibus and allows the user to talk with the microprocessor. After prompting for a Project-Programmer Number (PPN), the microprocessor expects a reply of 10 characters or less: two numbers (up to four digits each) separated by a comma and terminated by a carriage return.

The tenth character is always taken to be a carriage return, regardless of what is typed. If anything other than these characters are typed, or if the format is incorrect, the reply is rejected and the prompting message reissued. The only exception concerns multiple commas. These will be accepted at this stage, but will result in an invalid PPN message after communication with the PDP-11/04.

If the PPN is acceptable, the microprocessor prompts for a Password (PW), and expects a reply of a character or less. These may be anything other than controls and should terminate with a carriage return. Again, the ninth character typed will always be taken to be a carriage return. This reply is not echoed. If a control is typed, the reply will be rejected, and the microprocessor reissues a prompt for the PPN.

If both PPN and PW are acceptable, the microprocessor attempts to verify them by communicating with the PDP-11/04. An ACK (actually the symbol @) is always sent before transmissions in either direction. This is used to correctly position the pointer to the data buffers and to reset the count. Except for the ACK, all information is sent and received in a 32-byte packet. If communication fails after three trials because of parity errors or time-out (the microprocessor waits for approximately one second), the user is allowed to log on unverified. If communication succeeds, the PDP-11/04 instructs the microprocessor whether or not to allow the user to log on. Permission may be denied because of an invalid PPN or PW in which case the user is instructed to try again, or because of time-of-day restrictions on certain projects. The system then prompts for another log on attempt. (If a user is denied permission to log on because of time restrictions, the next user, assuming power has not been turned off in the meantime, may find that under certain circumstances a valid PPN is rejected by the PDP-11/04. This occurs because the microprocessor

has failed to clear the buffer area. However, the rejection of the PPN causes the buffers to be cleared and subsequent log on attempts will be correctly processed.)

If permission to log on is granted to the user, the microprocessor prints the date and time of the logging on as received from the PDP-11/04, and then releases the Unibus of the user's PDP-11. This machine will then execute the start command which initiated the interrupt of the microprocessor, and begin execution of the program at the starting address.

Once the user has logged on, the microprocessor halts, waiting for an interrupt which will occur when the user logs off. When this occurs, the microprocessor again assumes control of the PDP-11 Unibus. It sends an ACK to the PDP-11/04 followed by a packet in which the only significant information is the character 'L' in the first byte. Three attempts are made to communicate with the PDP-11/04. If it succeeds, the microprocessor finds in 'L' as the first character of the response packet. It then prints out the date, time and the time elapsed since log on, all as received from the PDP-11/04, followed by a log off message. Only this message is printed if communication fails. In any event, the user is logged off and the microprocessor retaining control of the PDP-11 Unibus, prompts for another log on. Until power is turned off, the SBC 80/10 continuously cycles through its log on and log off procedures.

2.3 PDP-11/04 Data Logging Hardware

The PDP-11/04 system is a standard Digital Equipment Corporation system with no modifications. The PDP-11/04 systems is configured as follows:

- | | |
|--------------------------|-------------------------------|
| . PDP-11/04 FC | 8K core memory computer |
| . DL-11WA | Clock and TTY Interface |
| . DZ-11E | 16 Channel RS232 Interface |
| . RX-11BA | Dual Drive Floppy Disc System |
| . LSI Lear-Siegler ADM-3 | CRT Terminal |

The system uses 15 of the RS232 channels on the DZ-11E for communication to the Microprocessor Interface Units located in the various PDP-11's in the laboratory. Thus the system could handle the monitoring function on 15 systems with no changes to the PDP-11/04 system. The communications is done at 4800 baud. The remaining communications channel is connected to the DEC-10 system for updating account numbers and passwords.

The PDP-11/04 can be tested using standard DEC diagnostic software which could be loaded from floppy disc. To verify proper operation of the software, it is possible to attach RS232 CRT terminal at any of the 15 RS232 ports in place of a Microprocessor Interface Unit. The PDP-11/04 does not time out on any transmissions thus it is possible to simulate the messages from the microprocessor interfaces and observe the PDP-11/04's response.

The PDP-11/04 system is a standard digital equipment configuration system with no modifications. The PDP-11/04 system is configured as follows:

8K core memory computer	PDP-11/04 FC
Clock and TTY interface	DE-110A
16 Channel RS232C interface	DE-110C
Small Format Floppy Disk System	RS-110A
CTT Terminal	LSI Term-2 (option) RM-3

The system uses 16 of the RS232C channels on the DE-110C for communication to the Microprocessor Interface Unit located in the various PDP-11's in the laboratory. Thus the system could handle the monitoring function as a system with no changes to the PDP-11/04 system. The communication is done at 4800 baud. The monitoring computer interface channel is connected to the PDP-11/04 system for updating system status and data.

2.4. Documentation for the Program ACCT

The PDP-11/04 can be tested using standard DEC diagnostic software which would be loaded from floppy disk. To monitor system operation of the software it is possible to attach RS232C CMT terminal at any of the 16 RS232C ports in place of a Microprocessor Interface Unit. The PDP-11/04 does not take on any transmission thus it is possible to simulate the messages from the microprocessor interface and observe the PDP-11/04 response.

The program ACCT is the part of the system which controls the PDP-11/04 when it is monitoring other PDP-11's. Pages 28 through 32 are copies of a Help file which gives instructions on how to operate the system. (Part III does not apply to ACCT). ACCT has the duty of watching all the serial lines connected to the microprocessors in the other PDP-11's. It must respond to each message sent to it. If the message is a log on, it must verify its correctness using a file of USERS account numbers and passwords and respond accordingly. If a log on is permitted it must make a record in the ENTRYYS file and put a message on the console. When the message is a log off the ENTRYYS record must be completed and messages must be sent to the corresponding microprocessor and the console. Pages 5 and 6 are a listing of a Help file which gives the general algorithms for this program.

THIS MONITORING SYSTEM RUNS UNDER THE DEC OPERATING SYSTEM RT-11

I. IN ORDER TO OPERATE THE SYSTEM IT IS NECESSARY TO BRING UP THE
RT-11 MONITOR.

1. TURN ON THE POWER.
2. PUT THE DISK LABELED "SYSTEM DISK" IN DRIVE #0
(THE LEFT DISK).
3. PUT THE DISK LABELED "USERS AND ENTRIES" IN DISK DRIVE #1
(THE RIGHT DISK).
4. PRESS THE BUTTON ON THE FRONT PANEL LABELED "CNTRL" AND WHILE
HOLDING IT DOWN PRESS THE BUTTON LABELED "BOOT".
5. RT-11 WILL ANSWER

RT-1153

V02C-02

YOU SHOULD THEN TYPE A STATE OF THE GENERAL FORM
DATE DD-MMM-YY

WHERE DD IS THE NUMBER OF THE DAY

MMM IS A THREE LETTER ABBREVIATION FOR THE MONTH

YY IS THE RIGHT TWO DIGITS OF THE YEAR

FOR EXAMPLE

DATE 21-JAN-78

6. RT-11 WILL ANSWER

YOU SHOULD THEN TYPE A STATEMENT OF THE GENERAL FORM
TIME HH:MM:SS

WHERE HH IS THE HOUR ON A 24 HOUR BASIS

MM IS THE MINUTE AFTER THE HOUR COUNT

SS IS THE SECOND COUNT

FOR EXAMPLE

TIME 14:52:27

THERE ARE TWO PROGRAMS AVAILABLE IN THIS SYSTEM: ACCT AND COM.

II. 'ACCT' IS THE PROGRAM WHICH MONITORS THE PDP-11'S IT SHOULD RUN AT ALL TIMES EXCEPT WHEN THE SYSTEM ADMINISTRATOR IS UPDATING FILES.

'ACCT' IS CALLED FROM RT-11 BY TYPING

R ACCT

THE SYSTEM THEN CONTINUES TO RUN, DISPLAYING LOGON'S AND LOGOFF'S ON THE CONSOLE, ONE CAN EXIT THIS SYSTEM AND RETURN TO RT-11 BY HITTING ^C (CONTROL C) TWICE ON THE KEYBOARD.

III. 'COM' IS A PROGRAM WHICH MAKES THE SYSTEM CONSOLE APPEAR TO BE IN TIMESHARING MODE ON THE DEC-10.

'COM' IS CALLED FROM RT-11 BY TYPING

R COM

AT THIS POINT THE SYSTEM ADMINISTRATOR MAY USE THE CONSOLE IN TIMESHARING MODE ON THE DEC-10, JUST AS ANY OTHER TERMINAL EXCEPT FOR THE FOLLOWING RULES:

1. C IS AN ESCAPE CHARACTER USED BY 'COM;', IN ORDER TO SEND A C TO THE DEC-10, HIT C C, ONLY THE SECOND OF THESE IS SENT TO THE DEC-10.
2. THE SEQUENCE C FOLLOWED BY ANY OTHER CHARACTER IS RESERVED FOR USE BY THE SYSTEM AND SHOULD NOT BE USED EXCEPT WHERE NOTED BELOW.
3. IN ORDER TO EXIT 'COM' HIT THE SEQUENCE
C E C C
THIS RETURNS CONTROL TO RT-11
4. WE CAN TRANSFER A USERS FILE FROM THE DEC-10 AS FOLLOWS:
LOGIN ON THE DEC-10 AND THEN TYPE
RUN FLOAD

THE SYSTEM WILL NOTIFY YOU WHEN THE TRANSFER IS COMPLETE
AND CONTROL IS RETURNED TO 'COM'.

5. WE CAN TRANSFER AN "ENTRYS" FILE IN THE DEC-10 AS FOLLOWS:

LOGIN ON THE DEC-10 AND THEN TYPE

RUN FACTS

THE SYSTEM WILL NOTIFY YOU WHEN IT IS DONE AND CONTROL IS
RETURNED TO 'COM'.

COMMENT: EACH LINE MONITORED BY THIS PROGRAM, ACCT, HAS TWO PROPERTIES,
A STATE AND A STATUS.

THE STATE IS EITHER UP OR DOWN AND IS MAINTAINED BY THE HARDWARE.
A VARIABLE IS USED TO REMEMBER ITS STATE UPON ITS LAST EXAMINATION.

THE STATUS OF EACH LINE IS LOGGED.ON OR LOGGED.OFF AND IS DETER-
MINED BY WHETHER A VARIABLE NACCBK IS 0 OR IS A POINTER TO AN
ENTRYS BLOCK WHICH CONTAINS A RECORD REFERRING TO THIS LINE.

EVERY MESSAGE SENT AND RECEIVED BY THIS PROGRAM IS AN ACK,
(@), FOLLOWED BY 32. CHARACTERS.

THERE ARE TWO TYPES OF MESSAGE RECEIVED BY THIS PROGRAM

- 1) THE LOGOFF MESSAGE WHICH STARTS WITH AN "L".
- 2) THE LOGON MESSAGE WHICH DOES NOT START WITH AN "L".

THERE ARE 3 TYPES OF MESSAGE TRANSMITTED BY THIS PROGRAM

- 1) THE RETURN MESSAGE TO A VALID LOGON MESSAGE WHICH
STARTS WITH A "V".
- 2) THE RETURN MESSAGE TO AN INVALID LOGON MESSAGE
WHICH STARTS WITH AN "F" AND "N", "P", OR "T" IN
THE SECOND LOCATION.
- 3) THE RETURN MESSAGE TO A LOGOFF MESSAGE WHICH
STARTS WITH AN "L".

THIS PROGRAM DOES NOT REQUIRE AN ACKNOWLEDGMENT FOR
ANY OF ITS MESSAGES. IT DOES NOT USE A TIMER FOR ANY OTHER
PURPOSE THAN TO MAINTAIN THE TIME OF DAY.

POLL.LINES:

REPEAT

FOR LINE.I :=1 TO NUMBER.OF.LINES

IF RECEIVE.BUFFER(LINE.I) HAS > 31. CHARACTERS THEN

CALL RECEIVED.A.MESSAGE

SEND THE RETURN MESSAGE TO LINE.I

END.IF

IF LINE.I WAS UP AND IS NOW DOWN THEN

CALL LOGOFF

SEND THE RETURN MESSAGE TO LINE.I

END.IF

END.FOR

UNTIL EXIT

```

RECEIVED.A.MESSAGE;
  IF MESSAGE STARTS WITH AN "L" OR LINE.I IS LOGGED.ON THEN CALL LOGOFF
  ELSE
    IF PPN FORMAT IS BAD THEN
      CALL BAD.PPN
    ELSE
      IF PASSWORD FORMAT IS BAD THEN
        CALL BAD.PW
      ELSE
        SEARCH USERS FILE FOR PPN
        IF NOT FOUND THEN
          CALL BAD.PPN
        ELSE
          IF PASSWORD IS WRONG THEN
            CALL BAD.PW
          ELSE
            IF TIME IS WRONG THEN
              CALL BAD.TIME
            ELSE
              CALL LOGON
            END.IF
          END.IF
        END.IF
      END.IF
    END.IF
  END.IF
END.IF
RETURN

```

LOGON

```

LOCATE AND READ THE NEXT AVAILABLE ACCOUNT BLOCK AND RECORD IN
  ENTRYS FILE
SAVE ACCOUNT BLOCK NUMBER IN NACCBK( LINE.I )
SAVE RECORD LOCATION IN NENTRY( LINE.I )
STORE PPN IN THE RECORD
STORE LINE.I IN THE RECORD AND THE CONSOLE MESSAGE
STORE DATE AND TIME IN THE LOGON AND LOGOFF FIELDS OF THE RECORD,
  IN THE RETURN MESSAGE, AND THE CONSOLE MESSAGE
INCREMENT THE NUMBER OF USED RECORDS IN THIS BLOCK
WRITE THE BLOCK BACK TO THE ENTRYS FILE
PRINT THE CONSOLE MESSAGE
PUT ACK, "V" IN THE FRONT OF THE RETURN MESSAGE
RETURN

```

LOGOFF:

```

  IF LINE.I IS LOGGED ON THEN
    READ THE ENTRYS BLOCK INDICATED BY NACCBK(LINE.I)
    GET A RECORD POINTER FROM NENTRY(LINE.I)
    STORE DATE AND TIME IN LOGOFF FIELD OF RECORD, IN
      RETURN MESSAGE AND CONSOLE MESSAGE
    STORE ELAPSED TIME IN RETURN MESSAGE AND CONSOLE MESSAGE
    WRITE BLOCK BACK TO THE ENTRYS FILE
    ZERO NACCBK(LINE.I)
    PRINT THE CONSOLE MESSAGE
    PUT ACK, "L" IN RETURN MESSAGE
  END.IF
RETURN

```

DZ11 Buffer and Message Handling

While the present DZ11 multiplexor configuration will support 16 full duplex asynchronous lines, the multiplexor is handled by 2 interrupt routines, one for input and one for output. As all lines may be transmitting and sending at the same time a good deal of the effort required in this program is directed in handling data in such a way the same code can handle all the lines.

The significant data structures related to DZ11 input, output are described next. Suppose that for a given transaction the line involved is LINE.J. For receiving there are three 1-dimensional arrays indexed by LINE.J. (In fact the arrays are words so that the index must be doubled).

RBUFA (LINE.J) is a pointer to a receive buffer area assigned permanently to LINE.J. (In fact it points 1 byte before the beginning of the buffer.) That area would be labelled RBUF(LINE.J) and is 34 bytes long.

RBUFP (LINE.J) is a pointer into RBUF (LINE.J) which points to the last character to be installed into that buffer.

RLEN (LINE.J) is a character count of the characters in RBUF (LINE.J). When this count reaches 32, the buffer is considered full and the message is delivered.

RBUFA is primarily used to initialize RBUFP. The characters are placed in the buffer by DZ11INT. The receipt of the character "@" on LINE.J always causes

$$\begin{array}{lcl} \text{RBUFA}(\text{LINE.J}) & \rightarrow & \text{RBUFP}(\text{LINE.J}) \\ 0 & \rightarrow & \text{RLEN}(\text{LINE.J}) \end{array}$$

which resets the buffer. This operation is also accomplished by DIALOG when it receives a message.

On transmission we also have a similar set of arrays which are distinct from those used for receiving. We have TBUFA (LINE.J) with the address of TBUF (LINE.J) which is 34 bytes long.

RBUF (LINE.J) is a pointer to the last character sent and TLEN (LINE.J) is the number of characters in the buffer which are not yet sent.

All transmissions are initialized by the routine SEND which initializes TBUF (LINE.J) and TLEN (LINE.J). SEND then arms the DZ11 to send characters on LINE.J. This causes interrupts to be serviced by DZOINT. DZOINT places characters on LINE.J adjusting TBUF and TLEN appropriately. When TLEN (LINE.J) = 0 then LINE.J is shut down.

Synchronization with the main program is accomplished as follows. When RLEN(LINE.J) > 31, then the main program handles the message.

When TLEN(LINE.J)=0 then the main program is free to initialize another transfer on LINE.J.

When a line which was in an active or up state goes down the hardware causes a bit to be clear which the main program detects and then processes as a log off.

The USERS File

The USERS file is initially loaded by the program COM and FLOAD. ACCT never writes in USERS. It simply uses the search technique described below to find the entry associated with the PPN generated by the log on message. The password from the log on message is checked against the password in the USERS entry. Also times for log on can be checked against the entry. Log on is then permitted or denied.

The USERS file is a data structure residing on the RX01 under RT-11. The exact file name is DX1:USERS.000. This structure is used to maintain records of all the users allowed on any of the monitored PDP-11's. The file is arranged so as to limit the search time for any given record. The technique used is a HASH function. The USERS file is 10 blocks long. These blocks are relative blocks 0,1,...,9 in the file. Each block contains a field called NEXTBL which is a pointer to the next block. Thus we have the circularly linked lists:

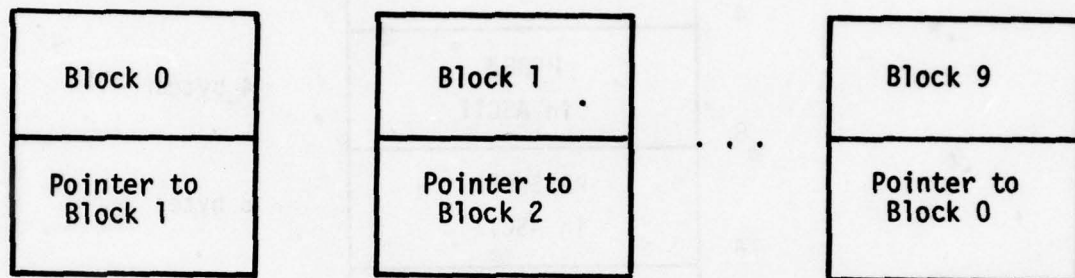


Figure 2.4.1 List Layout

Each block contains space for 11 user specification records, each of which is 46 bytes long. Thus the layout of a block is

byte	
0.-45.	User record #1
46.-91.	User record #2
.	.
.	last occupied user record
.	unused records
506.-507.	NUMUSR-number of user records in this block
508.-509.	NEXTBL
510.-511.	unused

Figure 2.4.2 Block Layout

In order to see where a new user record is entered and how we search for it, we need to examine the hash function. Take the last 2 digits of the programmer number and compute the value modulo 10. That is a

number in the list 0,1,...,9. The respective block in USERS is the first choice of where to put this user record on entry; or to search for it if we wish to find it. If that block is full we move to the next block in the list, etc. 109 user records is the maximum number that the system will handle

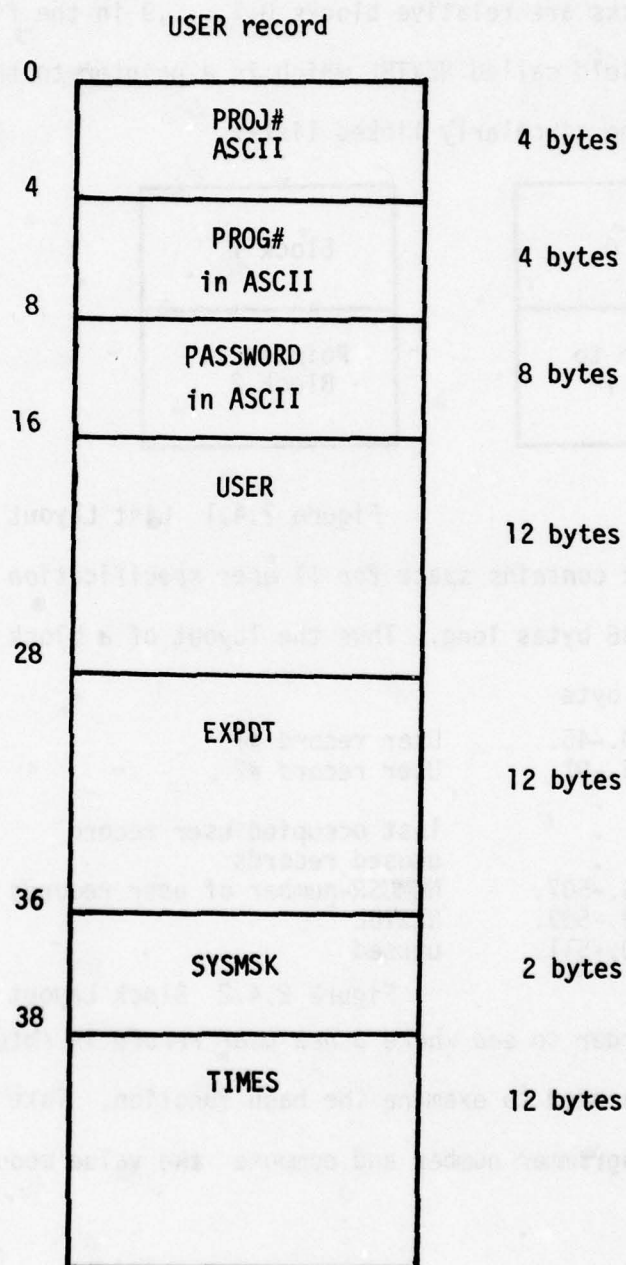


Figure 2.4.3 User Record

The ENTRYS File

The ENTRYS file has a first block which although 512 bytes long contains only a significant first word. It is the number of the first block in the file which has space for a new accounting entry. All other blocks are arranged.

Byte

0-29	account record #1
	:
	last full account record
	:
480-509	account record #17
510-511	number of records in this block

Each record is arranged

0-1	PDP-11 #
2-5	Log on date
6-9	Log on time
10-13	Log off date
14-17	Log on time
18-29	PPN

All time and date formats are RT-11 V.2 standards and can be obtained from RT-11 manuals. Whenever possible processing is done by RT-11 system subroutines.

Whenever a log on is permitted on LINE.J the block #0 is read and the block of ENTRYS with the next open record is thus located. This entry is then claimed, filled in and written back on the disk. A record of this transaction is kept in core by two variables. NACCBK(LINE.J) retains the block number of this entry. This variable is set to zero when this line is logged off so it serves as a flag to show the status of the line. Also NENTRY(LINE.J) is retained as a pointer to the position of the record in the buffer ACCBUF (which is used for all block transfers into and out of the ENTRYS file). Thus when this line requests a log off NACCBK(LINE.J) is consulted. Then the appropriate block is read and updated and is then written back onto the disk.

2.5 Documentation for the Program COM

TABLE OF CONTENTS

Note: File names are enclosed in (). Data Structures are enclosed in < >.
Program names are enclosed in [].

[COM]

I. [MONITOR] (SMON)

<PROCESS CONTROL BLOCKS>

<QUEUES>

[GET]

[PUT]

<READY QUEUE>

<SEMAPHORES>

[P]

[V]

[SW]

II. PROCESSES (CMAST)

A. [MASTER]

INITIALIZE

IDLE PROCESS

B. [TTIP]

TERMINAL-IN CONTROL

<CHARACTER BUFFERS>

<TTIQ>

<TTIQS>

DEC-10 LINE-OUT CONTROL

<DZOQ>

TTIP STATE DIAGRAM

C. [DZIP]

DEC-10 LINE-IN CONTROL

<DZIQ>

<DQIQS>

TERMINAL-OUT CONTROL

<TTOQ>

DZIP STATE DIAGRAM

III. INTERRUPT ROUTINES (CINU)

- A. [TTIINT]
- B. [TTOINT]
- C. [DZ2IINT]
- D. [DZ2OINT]

IV. APPLICATIONS PROGRAMS

A. RECEIVE A USERS SPECIFICATIONS FILE FROM THE DEC-10

- 1. [RECAFILE](ROF)
 - <USERS FILE>
 - <USERS RECORDS:PDP-11,DEC-10>

- 2. [REC.RT] (ROF)
- 3. [FLOAD](DEC-10 FILE FLOAD)

B. SEND AN ACCOUNTING ENTRYS FILE TO THE DEC-10

- 1. [SENDAFILE](SOF)
- 2. [ACK,RT](SOF)
- 3. [FACTS](DEC-10 FILE FACTS)

C. TRANSFER A FILE TO THE DEC-10 FOR DEVELOPMENT PURPOSES

- 1. [TRANF](SOF)

[COM]

COM is a program to manage communications between the PDP-11/04 and the DEC 10. These two machines are connected by a 1200 baud asynchronous full duplex serial line. To the DEC 10, this line is interfaced in such a way that the PDP-11/04 appears to be a timesharing terminal. Thus it is the responsibility of the PDP-11/04 and its operator to generate the proper timesharing dialog. The method chosen to do this is transparency. Thus

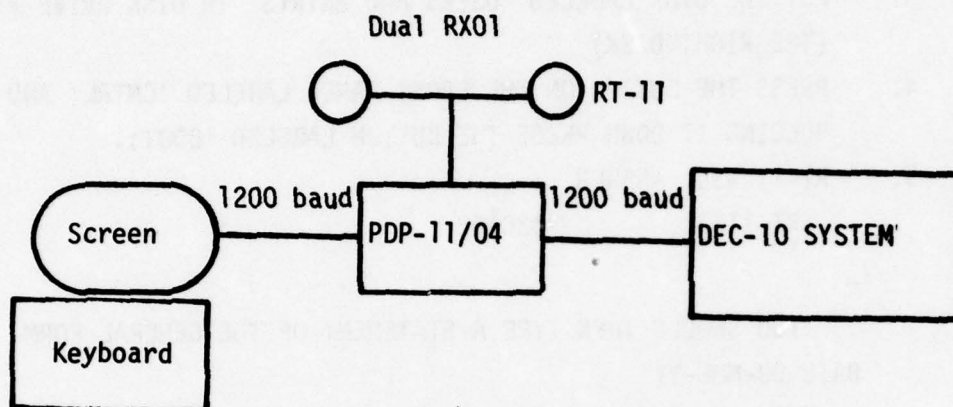


Figure 2.5.1 System Configuration

appears to be

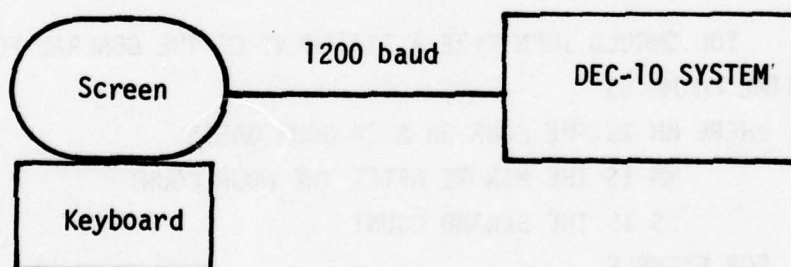


Figure 2.5.2 Apparent System Configuration

most of the time. The operator instructions are as follows. (Part II does not apply to COM).

THIS MONITORING SYSTEM RUNS UNDER THE DEC OPERATING SYSTEM RT-11

I. IN ORDER TO OPERATE THE SYSTEM IT IS NECESSARY TO BRING UP THE RT-11 MONITOR.

1. TURN ON THE POWER.
2. PUT THE DISK LABELED 'SYSTEM DISK' IN DRIVE #0 (THE LEFT DISK).
3. PUT THE DISK LABELED 'USERS AND ENTRIES' IN DISK DRIVE #1 (THE RIGHT DISK).
4. PRESS THE BUTTON ON THE FRONT PANEL LABELED 'CNTRL' AND WHILE HOLDING IT DOWN PRESS THE BUTTON LABELED 'BOOT';.
5. RT-11 WILL ANSWER
RT-11SJ V02C-02

—
YOU SHOULD THEN TYPE A STATEMENT OF THE GENERAL FORM
DATE DD-MMM-YY

WHERE DD IS THE NUMBER OF THE DAY

MMM IS A THREE LETTER ABBREVIATION FOR THE MONTH

YY IS THE RIGHT TWO DIGITS OF THE YEAR

FOR EXAMPLE

DATE 21-JAN-78

6. RT-11 WILL ANSWER

—
YOU SHOULD THEN TYPE A STATEMENT OF THE GENERAL FORM
TIME HH:MM:SS

WHERE HH IS THE HOUR ON A 24 HOUR BASIS

MM IS THE MINUTE AFTER THE HOUR COUNT

SS IS THE SECOND COUNT

FOR EXAMPLE

TIME 14:52:27

THERE ARE TWO PROGRAMS AVAILABLE IN THIS SYSTEM: ACCT AND COM.

- II. 'ACCT' IS THE PROGRAM WHICH MONITORS THE PDP-11'S. IT SHOULD RUN AT ALL TIMES EXCEPT WHEN THE SYSTEM ADMINISTRATOR IS UPDATING FILES.

'ACCT' IS CALLED FROM RT-11 BY TYPING

R ACCT

THE SYSTEM THEN CONTINUES TO RUN, DISPLAYING LOGON'S AND LOGOFF'S ON THE CONSOLE. ONE CAN EXIT THIS SYSTEM AND RETURN TO RT-11 BY HITTING C (CONTROL C) TWICE ON THE KEYBOARD.

- III. 'COM' IS A PROGRAM WHICH MAKES THE SYSTEM CONSOLE APPEAR TO BE IN TIMESHARING MODE ON THE DEC-10.

'COM' IS CALLED FROM RT-11 BY TYPING

R COM

AT THIS POINT THE SYSTEM ADMINISTRATOR MAY USE THE CONSOLE IN TIMESHARING MODE ON THE DEC-10, JUST AS ANY OTHER TERMINAL EXCEPT FOR THE FOLLOWING RULES:

1. C IS AN ESCAPE CHARACTER USED BY 'COM', IN ORDER TO SEND A C TO THE DEC-10, HIT C C. ONLY THE SECOND OF THESE IS SENT TO THE DEC-10.
2. THE SEQUENCE C FOLLOWED BY ANY OTHER CHARACTER IS RESERVED FOR THE USE BY THE SYSTEM AND SHOULD NOT BE USED EXCEPT WHERE NOTED BELOW.
3. IN ORDER TO EXIT 'COM' HIT THE SEQUENCE
C E C C
THIS RETURNS CONTROL TO RT-11.
4. WE CAN TRANSFER A USERS FILE FROM THE DEC-10 AS FOLLOWS:
LOGIN ON THE DEC-10 AND THEN TYPE
RUN FLOAD
THE SYSTEM WILL NOTIFY YOU WHEN THE TRANSFER IS COMPLETE AND CONTROL IS RETURNED TO 'COM'.

5. WE CAN TRANSFER AN 'ENTRYS' FILE TO THE DEC-10 AS FOLLOWS:
LOGIN ON THE DEC-10 AND THEN TYPE
RUN FACTS
THE SYSTEM WILL NOTIFY YOU WHEN IT IS DONE AND CONTROL IS
RETURNED TO 'COM'.
6. FLOAD CLEARS THE 'ENTRYS' FILE SO WHEN UPDATING THE SYSTEM
ALWAYS RUN 'FACTS FIRST AND THEN RUN 'FLOAD'!!!!!!!!!!

Part I is all dialog with the RT-11 operating system on the PDP-11/04. Part III begins in the same way. The command R COM causes the loading of the program COM which immediately enters the transparent mode. We see that COM is not completely transparent because we wish to do other functions besides maintain operator dialog with the DEC 10.

The two functions necessary to implement the monitoring systems are to load a Users specification file from the DEC 10 to a RT-11 disk file on the RX01; and to send a file of Entrys which contains the logged data from the monitored systems from an RT-11 file to the DEC-10.

In addition we have implemented an exit sequence and although the operator is not aware of its existence there is a general program to copy a file from the RX01 to the DEC 10. This sequence is used for program development and maintenance. In particular, the system programmer can obtain listings in this fashion. (In order to use this function, see page 35.)

The program COM is a multiple task or process program. It contains 3 processes.

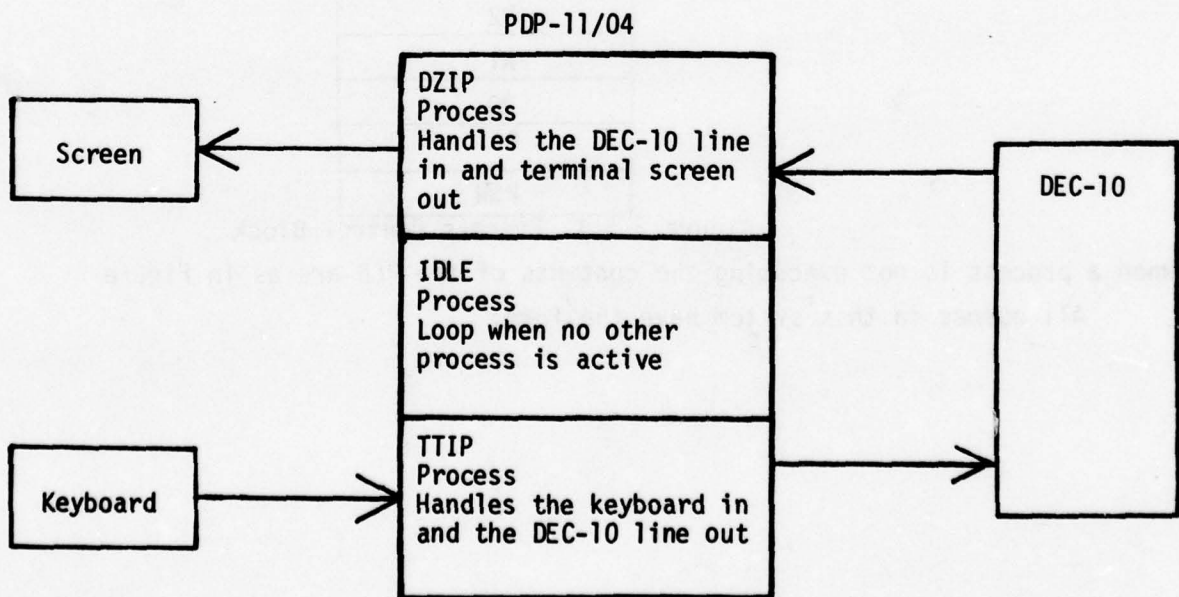


Figure 2.5.3 COM Process Diagram
45

The DEC-10 handles its timesharing terminals by echoing its character input from the CPU front end rather than in the terminal or modem. This allows greater flexibility in that certain characters may receive a 2 character echo or no echo at all. This is what terminal manufacturers call "full duplex" mode although all low speed lines are full duplex. Thus the processes of input and output from the terminal are independent and concurrent.

A small monitor was written to handle these concurrent processes. It also handles queuing of buffers and process synchronization. This program resides in the file SMON.

I. MONITOR (file SMON)

Each process in this system has a Process Control Block (PCB)

PCB:

addresses: ↓

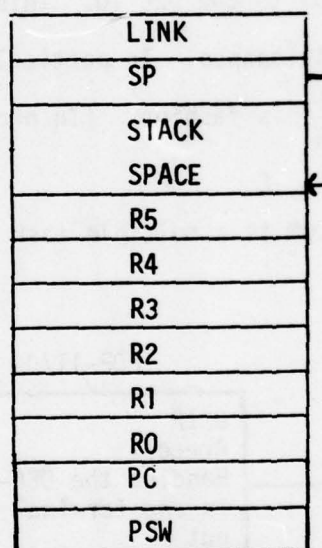


Figure 2.5.4 Process Control Block

When a process is not executing the contents of the PCB are as in Figure
All queues in this system have the form:

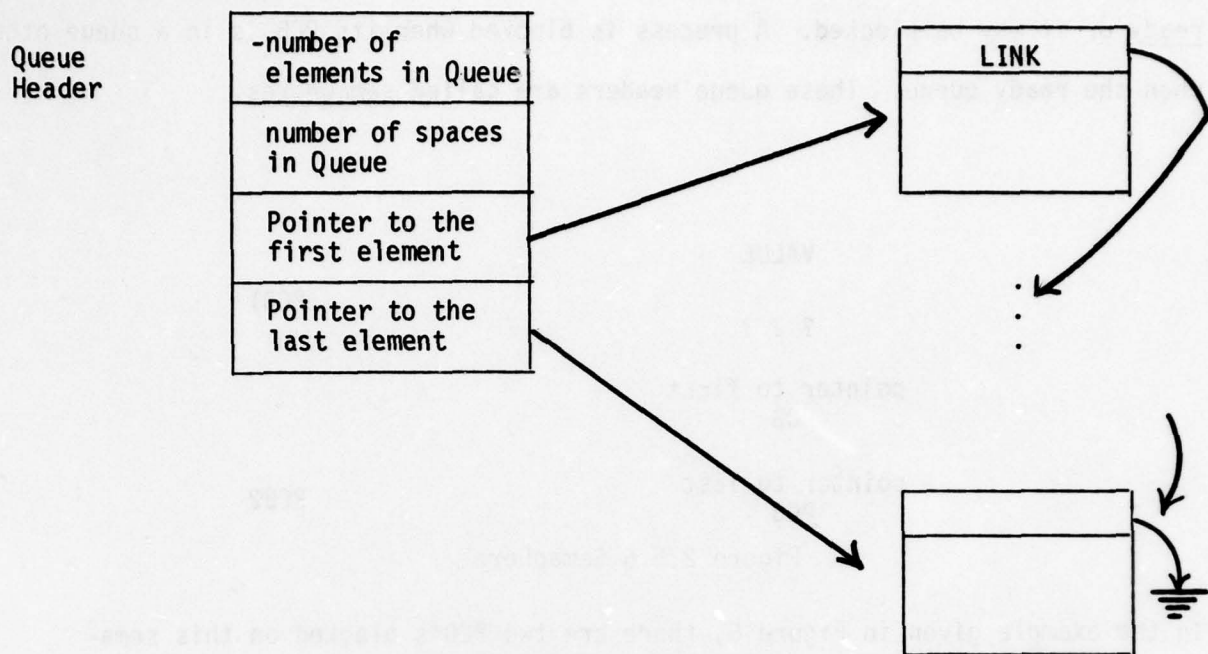


Figure 2.5.5 Queue Header

There are two things to note about this system.

- a) The size of the element in the queue is unimportant.
- b) The number of spaces in the queue is for control purposes and is not vital as the queue is really only a linked list of space which was obtained elsewhere.

There are two routines

GET(pointer to Queue Header, pointer to Element)

PUT(pointer to Queue Header, pointer to Element)

used to maintain these queues. The first argument, passed in R4, is always input; the second argument, passed in R5, is output by GET and is input to PUT). PUT sets R4 to zero on success. Get sets R4 to zero on failure.

There is a Monitor queue with a header whose global address is READY. It is a queue of PCB's that are ready to run but are not now running. There is a global pointer in the Monitor whose global address is CP. It always

points to the current running process. A process may be running, it may be ready or it may be blocked. A process is blocked when its PCB is in a queue other than the ready queue. These queue headers are called semaphores.

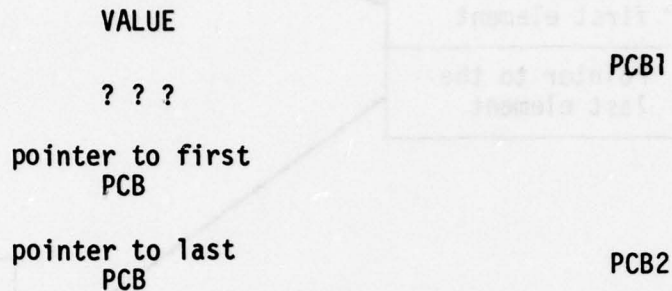


Figure 2.5.6 Semaphore

In the example given in Figure 6, there are two PCB's blocked on this semaphore so the VALUE = -2. For a normal queue header the value is the negative of the number elements in the queue. Thus acceptable values are 0, -1, -2 ..., -maximum number of elements in the queue. For a semaphore, we generalize this so that when the queue is empty, in addition to 0, the value may also be 1, 2, 3, ..

We further define two operations that a process may execute.

P(pointer to a semaphore), argument in R4
 V(pointer to a semaphore), argument in R4

P: Decrement the value of the semaphore by 1.

If the value is negative then

place the PCB of this process on this semaphore queue and
 make the first process on the ready queue the current process.

else

return

V: Increment the value of the semaphore by 1.

If the value is 0 or negative then

remove the first process from this semaphore queue and place
 it at the end of the ready queue.

return

These operations are used to signal between processes and between processes and interrupt routines. Interrupt routines always execute on the stack of the current process. For that reason, although they may execute a V which may make another process ready, they may not execute a P, for that would possibly block a routine which had nothing to do with that particular interrupt.

The monitor has one further operation, SW, called a short wait which simply places the current process at the end of the ready queue.

The implementation of the MONITOR is done with the TRAP instruction. Each MONITOR instruction operates at LEVEL 7 so that it is uninterruptable.

The MONITOR has no 'initialization' routine itself so that the main program must initialize all PCB's, the ready queue, and the current process.

II. PROCESSES

File CMAST

On the following page is a general description of this file as indicated in the help file. In more detail we have

MASTER (file CMAST)

MASTER: is the entry point of COM. Its first job is to initialize all queues, PCB's, devices and then it becomes the idle process. Thus it is always running, or it is on the ready queue. Master starts two other processes.

TTIP (file CMAST)

TTIP is the process which handles input from the TT. It also handles outputs on a DZ11 serial multiplexor. In this case only line #15 which is line #7 on the second DZ11 is used.

Input on the TT is always from a queue TTIQ. The input is put there by an interrupt routine, TTIINT (file CINU).

CMAST 18 THE MASTER ROUTINE FOR COM. IT FIRST TAKES CONTROL FROM RT-11, INITIALIZES THE INTERRUPT VECTORS, SETS UP ALL QUEUES AND PROCESS CONTROL BLOCKS. THE INTERRUPT HANDLERS ARE IN FILE CINU. THE QUEUE AND PROCESS HANDLING ARE DONE BY PROGRAMS GET, PUT, P, V, SV WHICH ARE REACHED BY THE TRAP INSTRUCTION AND ARE IN FILE SMON. CMAST THEN SETS UP TWO PROCESSES NAMED TTIP AND DZIP.

TTIP HANDLES ALL INPUT FROM THE CONSOLE DEVICE. IT ALSO TAKES CARE OF ALL OUTPUT ON THE DZ11 (I.E. TO THE DEC-10).

DZIP HANDLES ALL INPUT FROM THE DZ11 (I.E. FROM THE DEC-10) AND ALL OUTPUT ON THE CONSOLE. IN NORMAL MODE THESE TWO PROCESSES ALLOW A TRANSPARENT OPERATION WITH THE CONSOLE APPEARING TO BE A TERMINAL ON THE DEC-10.

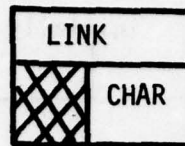
I. TTIP HAS SEVERAL SPECIAL FEATURES OTHER THAN TRANSPARENCY.

1. ^C IS USED AS AN ESCAPE CHARACTER. R SINGLE C INPUT ON THE CONSOLE IS NEVER SENT TO THE DEC-10. IT JUST CAUSES THE NEXT CHAR TO BE EXAMINED IN A DIFFERENT LIGHT.
2. IF THE NEXT CHAR IS A ^C OR ANY OTHER CHAR WITHOUT A SPECIFIC PURPOSE MENTIONED BELOW IT IS SENT TO THE DEC-10.
3. IF THE NEXT CHAR IS A ^E THEN COM SURRENDERS CONTROL OF THE CONSOLE TO RT-11 WHERE A ^C^C WILL RESTORE THE RT-11 MONITOR AND THE EXIT FROM COM IS COMPLETE.
4. IF THE NEXT CHAR IS A ^R THEN THE FILE DX1:DEC10.LST IS TRANSFERRED TO THE DEC-10 LINE. THE PROGRAM WHICH DOES THIS IS TRANF AND IT IS IN THE FILE ROF. TO USE THIS PREPARE THE DEC-10 BY CALLING ITS PIP AND SETTING UP A TRANSFER FROM TT: TO ANY FILE. YOU WILL SEE THE FILE GO AS IT IS ECHOED ON THE CONSOLE. WHEN IT IS DONE HIT A CR FOLLOWED BY A ^Z WHICH WILL CLOSE THE FILE.
4. IF THE NEXT CHAR IS A ^S THEN TTIP ENTERS A MODE WHICH SENDS THE FILE DX1:ENTRYS TO THE DEC-10 IN A SPECIAL FORMAT WHICH IS ERROR CHECKED. THE PROGRAM NAME IS SENDAFILE ON FILE SOF. THIS SEQUENCE SHOULD NEVER BE ENTERED FROM THE CONSOLE IT IS FAKED BY THE PROGRAM IN RESPONSE TO AN RED FROM THE DEC-10 PROGRAM FACTS.
5. AT ANY TIME THE CHAR ^Y IS ILLEGAL. IT IS USED AS AN ENQ TO CAUSE TTIP TO ENTER A MODE IN WHICH IT SENDS ACKS AND NACKS TO THE DEC-10 WHEN THE PROGRAM FLOAD IS SENDING A 'USERS' FILE TO PDP11. THE PROGRAM WHICH DOES THIS IS REC.RT AND IT IS IN FILE ROF.

II. DZIP IS ALSO DIVERTED FROM TRANSPARENCY IN SEVERAL CASES.

1. IT LOOKS FOR A REQ FROM THE DEC-10. WHEN IT GETS THIS CHAR IT SIGNALS TTIP WITH A ^S^S AND THEN ENTERS ACK.RT WHICH LOOKS FOR ACKS AND RACKS AND AN EOT. ACK.RT IS IN FILE SOF.
2. IT LOOKS FOR AN ENQ FROM THE DEC-10. WHEN IT GETS THIS CAR IT ENTERS A MODE WHICH RECEIVES A 'USERS' FILE FROM THE DEC-10 PROGRAM FLOAD THIS PROGRAM IS RECAFILE AND IS IN FILE ROF.

The elements of this queue each contain a single character



The input character queue is guarded by a semaphore TTIQS. TTIQS is initially 0. Each time TT11NT puts a character in the queue TTIQ it does a V(TTIQS). Thus TTIQS has the value of the number of characters ready to be processed. TTIP does a P(TTIQS) each time before it removes a character from the QUEUE. Thus if there are no characters and TTIP requests one it becomes blocked.

When TTIP has a character it normally puts it on another queue DZOQ and starts the DZ11 on line #15. The interrupt routine DZOINT then processes the character from the queue and sends it to the DEC-10.

The operation of TTIP as just described is completely transparent. The nontransparent features of TTIP are more complicated and are best described by a finite state machine.

TTIP state diagram

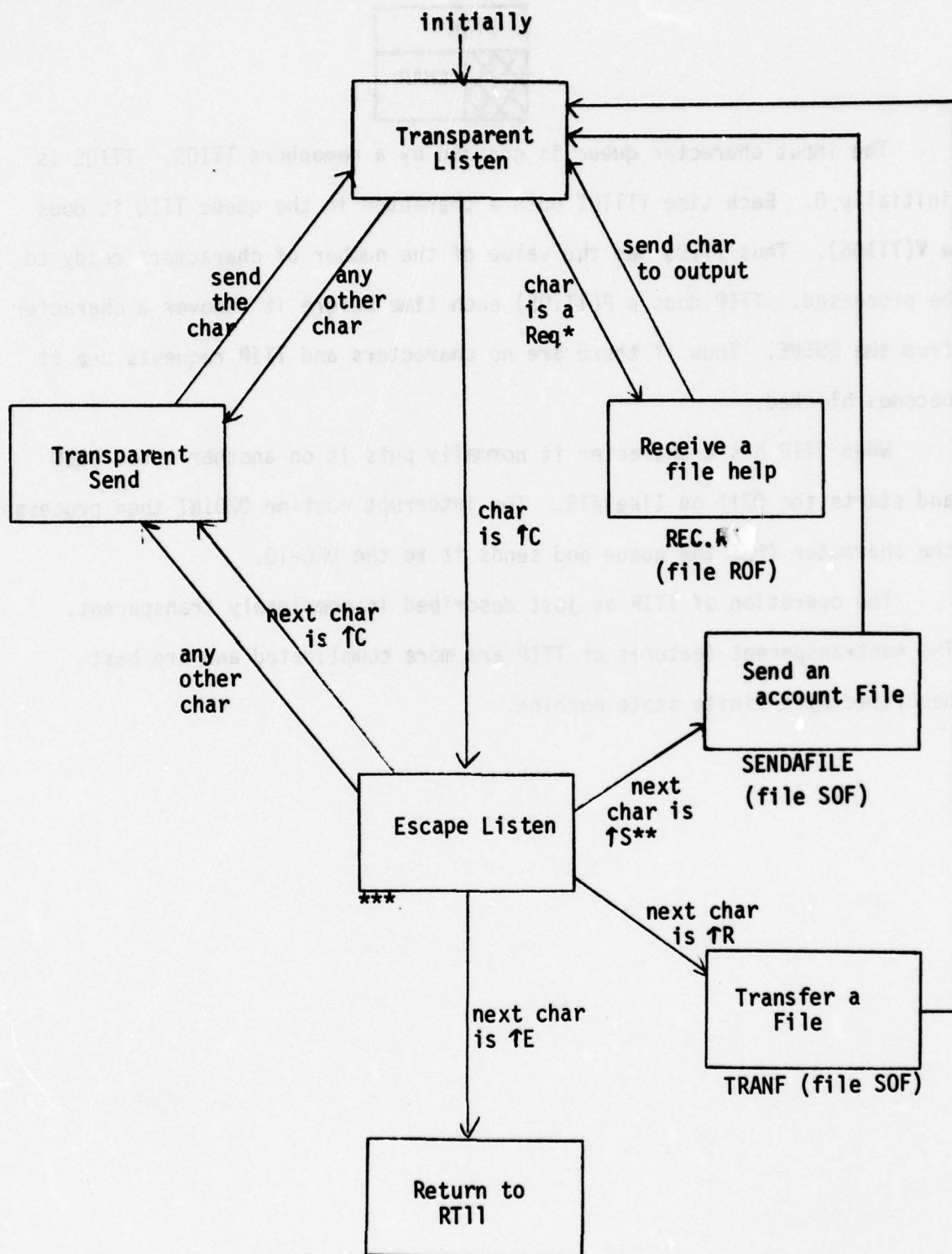


Figure 2.5.7 TTIP State Diagram.

- * REQ is a ASCII code not generated by the terminal. It is used internally as a signal from the routine RECEIVE-A-FILE in process DZIP to signal this process to help in the receiving of a USER file. This process will send acknowledgements to the DEC 10 which is part of the error checking process.
- ** The sequence +C +S should never be generated from the terminal. It is used internally by the routine DZIP to signal that this process should send the account file (ENTRIES) to the DEC 10.
- *** A variable CHRSAV is used to save the states TRANSPARENT LISTEN or ESCAPE LISTEN.

If CHRSAV \neq +C then the state is TRANSPARENT LISTEN

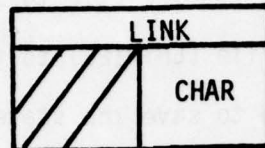
If CHRSAV = +C then the state is ESCAPE LISTEN.

DZIP (file CMAST)

DZIP is the process which handles input from the DEC-10 on DZ11 line 15. It also handles output on the TT.

Input on the DZ11 is always from a queue DZIQ. The input is put there by a routine DZIINT(file CINU).

The elements of this queue each contain a single character.



The input character queue is guarded by a semaphore DZIQS. DZIQS is initially 0. Each time DZIINT puts a character in the queue DZIQ it does a V(DZIQS). Thus DZIQS has the value of the number of characters to be processed. DZIP does a P(DZIQS) each time before it removes a character from the queue. Thus if there are no characters and DZIP requests one it becomes blocked.

When DZIP has a character it normally puts it on another queue TTOQ and starts the TT out interface. The interrupt routine TTOINT then processes the character from the queue and puts it out on the terminal.

The operation of DZIP as just described is completely transparent. The nontransparent features of DZIP are more complicated and are best described by a finite state machine.

DZIP State Diagram

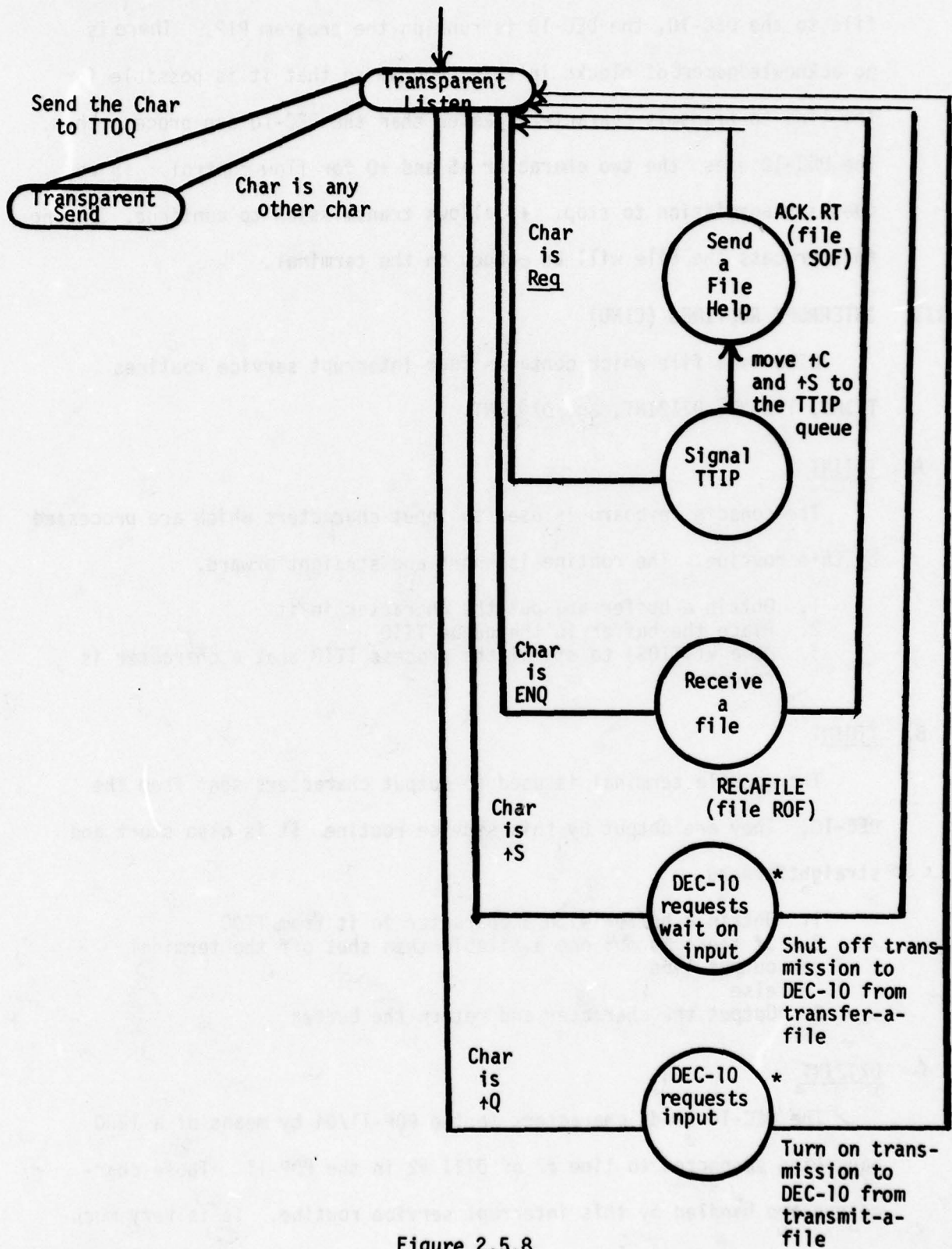


Figure 2.5.8

* During the time when transmit a file is sending characters from an RX01 file to the DEC-10, the DEC-10 is running the program PIP. There is no acknowledgement of blocks in this program so that it is possible for the line to transmit characters faster than the DEC-10 can process them. The DEC-10 uses the two character +S and +Q for flow control. +S requests transmission to stop. +Q allows transmission to continue. During this process the file will be echoed on the terminal.

III. INTERRUPT ROUTINES (CINU)

CINU is a file which contains four interrupt service routines TTINT, TTOINT, DZI2INT, and DZ02INT.

A. TTIINT

The console keyboard is used to input characters which are processed by this routine. The routine is short and straightforward.

1. Obtain a buffer and put the character in it
2. Place the buffer in the queue TTIQ
3. Do a V(TTIQS) to signal the process TTIP that a character is ready

B. TTOINT

The console terminal is used to output characters sent from the DEC-10. They are output by this service routine. It is also short and straightforward.

1. Obtain a buffer with a character in it from TTOQ
2. If there is not one available then shut off the terminal output line
else
3. Output the character and return the buffer

C. DZI2INT

The DEC-10 sends characters to the PDP-11/04 by means of a 1200 baud line connected to line #7 of DZII #2 in the PDP-11. Those characters are handled by this interrupt service routine. It is very much

like TTIINT.

1. Obtain a buffer and put the character in it
2. Place the buffer in the queue TTIQ
3. Do a V(DZIQS) to signal the process DZIP that a character is ready

D. DZ02INT

The PDP-11/04 sends character to the DEC-10, also over line #7 of DZII #2. Those characters are handled by this interrupt service routine which is very similar to TTIINT.

1. Obtain a buffer with a character in it from DZ0Q
2. If there is not one available then shut off the output line to the DEC 10
else
3. Output the character and return the buffer

IV. Applications Programs

There are three applications programs.

- A. Receive a User Specification File from the DEC-10
- B. Send an Accounting Entries File to the DEC-10
- C. Transfer a file to the DEC-10 for development purposes

Each applications program requires three processes to accomplish its function. Two are in the PDP-11/04 and one is in the DEC 10. The two in the PDP-11/04 are always allocated, one to run under TTIP and one under DZIP. Relative to the numbering above the programs are

	TTIP	DZIP	DEC 10
A.	REC.RT	RECAFILE	FLOAD
B.	SENDAFILE	ACK.RT	FACTS
C.	TRANF	DZIP	PIP

A. Receive a User Specification file from the DEC-10.

1. RECAFILE (file ROF)

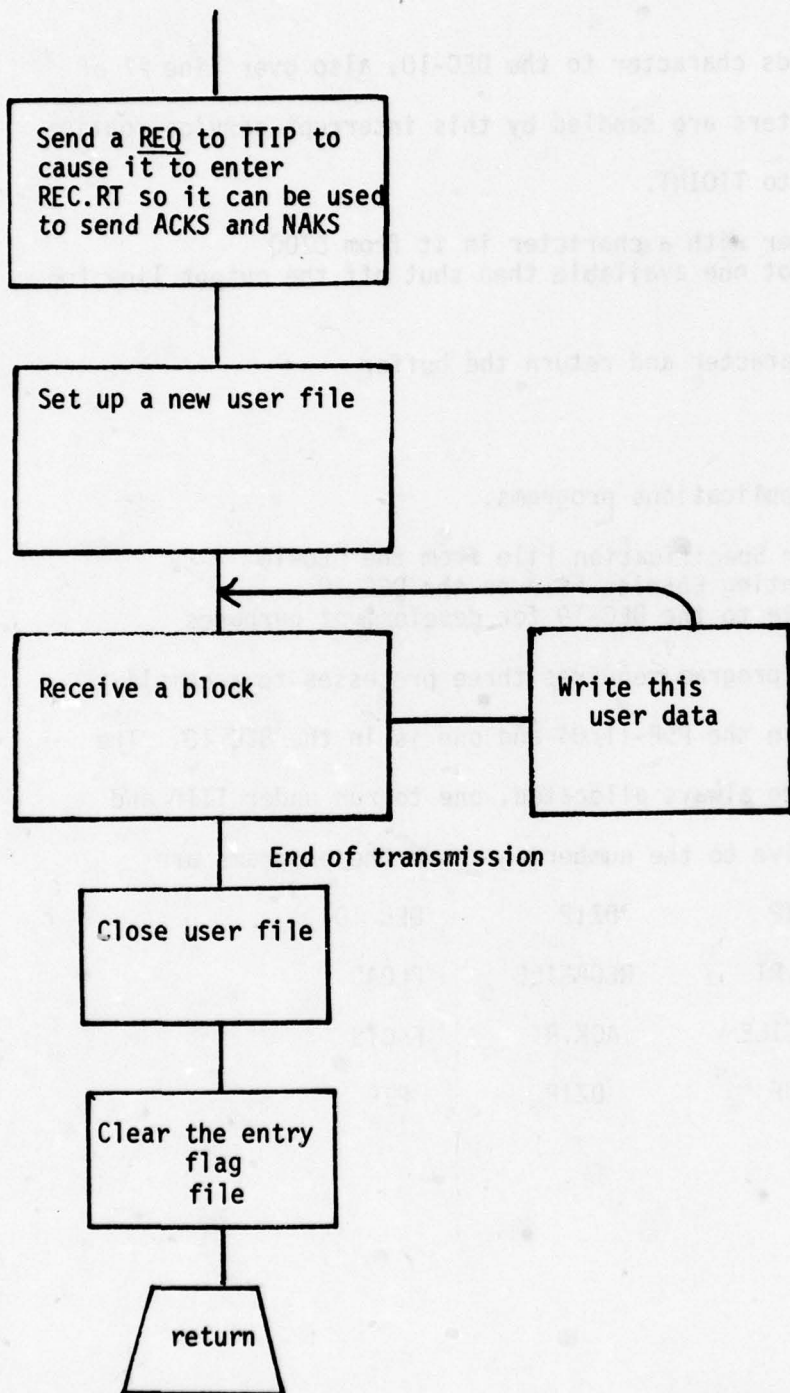
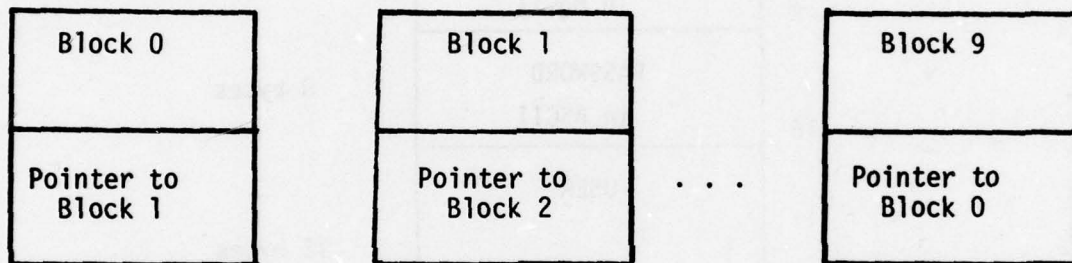


Figure 2.5.9 RECAFILE

The USERS file

The USERS file is a data structure residing on the RX01 under RT-11. The exact file name is DX1:USERS.~~000~~. This structure is used to maintain records of all the users allowed on any of the monitored PDP-11's. The file is arranged so as to limit the search time for any given record. The file is arranged so as to limit the search time for any given record. The technique used is a HASH function. The USERS file is 10 blocks long. These blocks are relative blocks 0,1,...,9 in the file. Each block contains a field called NEXTBL which is a pointer to the next block. Thus we have the circularly linked lists:



Each block contains space for 11 user specification records, each of which is 46 bytes long. Thus the layout of a block is

byte	
0.-45.	User record #1
46.-91.	User record #2
.	
.	last occupied user record
.	unused records
506.-507.	NUMUSR-number of user records in this block
508.-509.	NEXTBL
510.-511.	unused

In order to see where a new user record is entered and how we search for it, we need to examine the hash function. Take the last 2 digits of the programmer number and compute the value modulo 10. That is a

number in the list 0,1,...,9. The respective block in USERS is the first choice of where to put this user record on entry; or to search for it if we wish to find it. If that block is full we move to the next block in the list, etc. 109 user records is the maximum number that the system will handle.

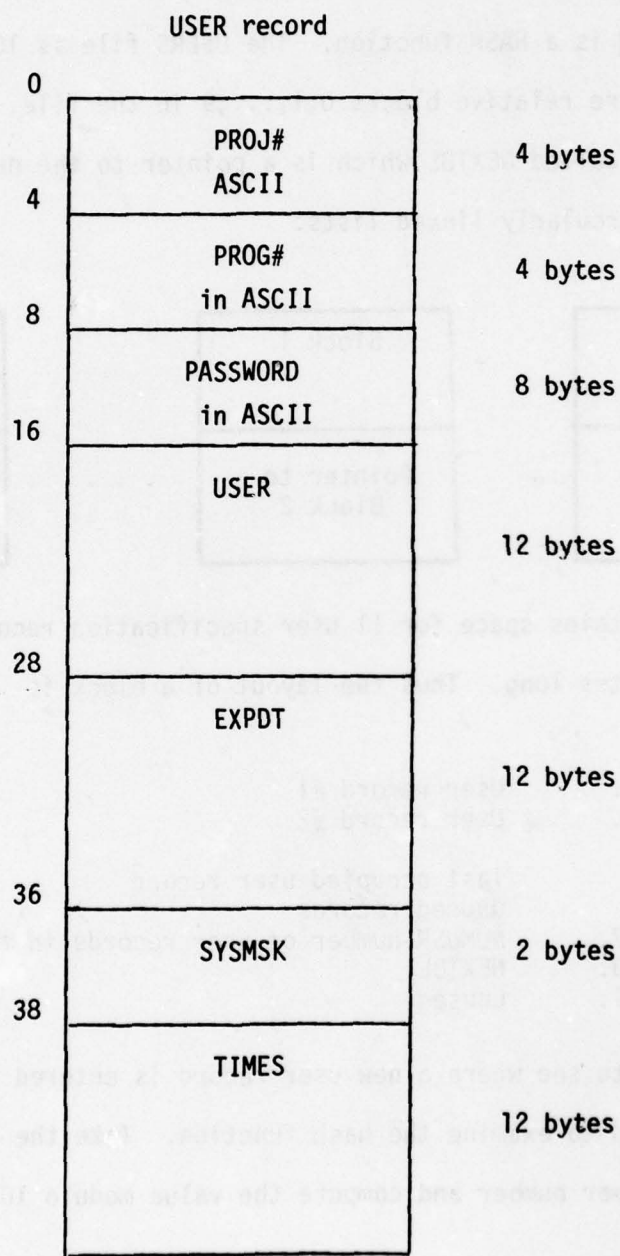


Figure 2.5.10 User Record
60

Now in order to set up a user file we do the following algorithm:

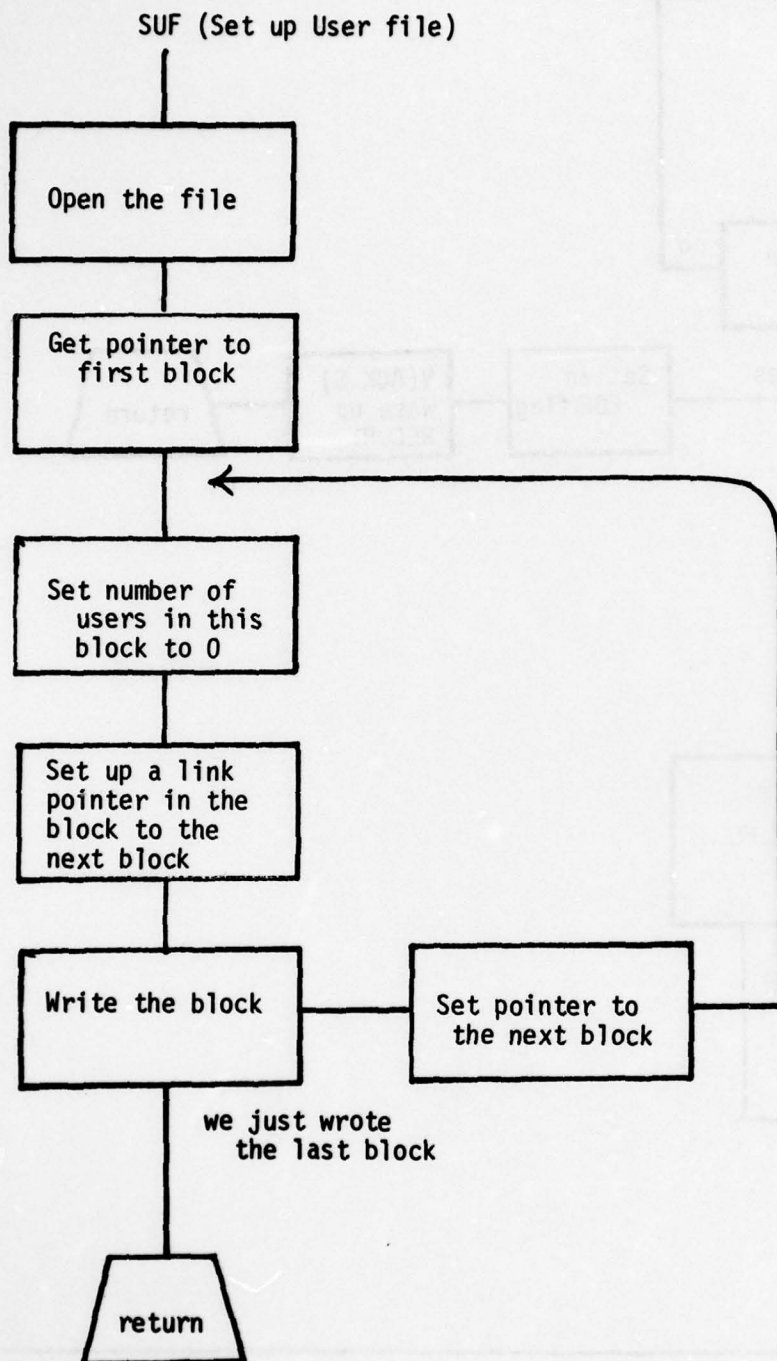


Figure 2.5.11 Set Up User File

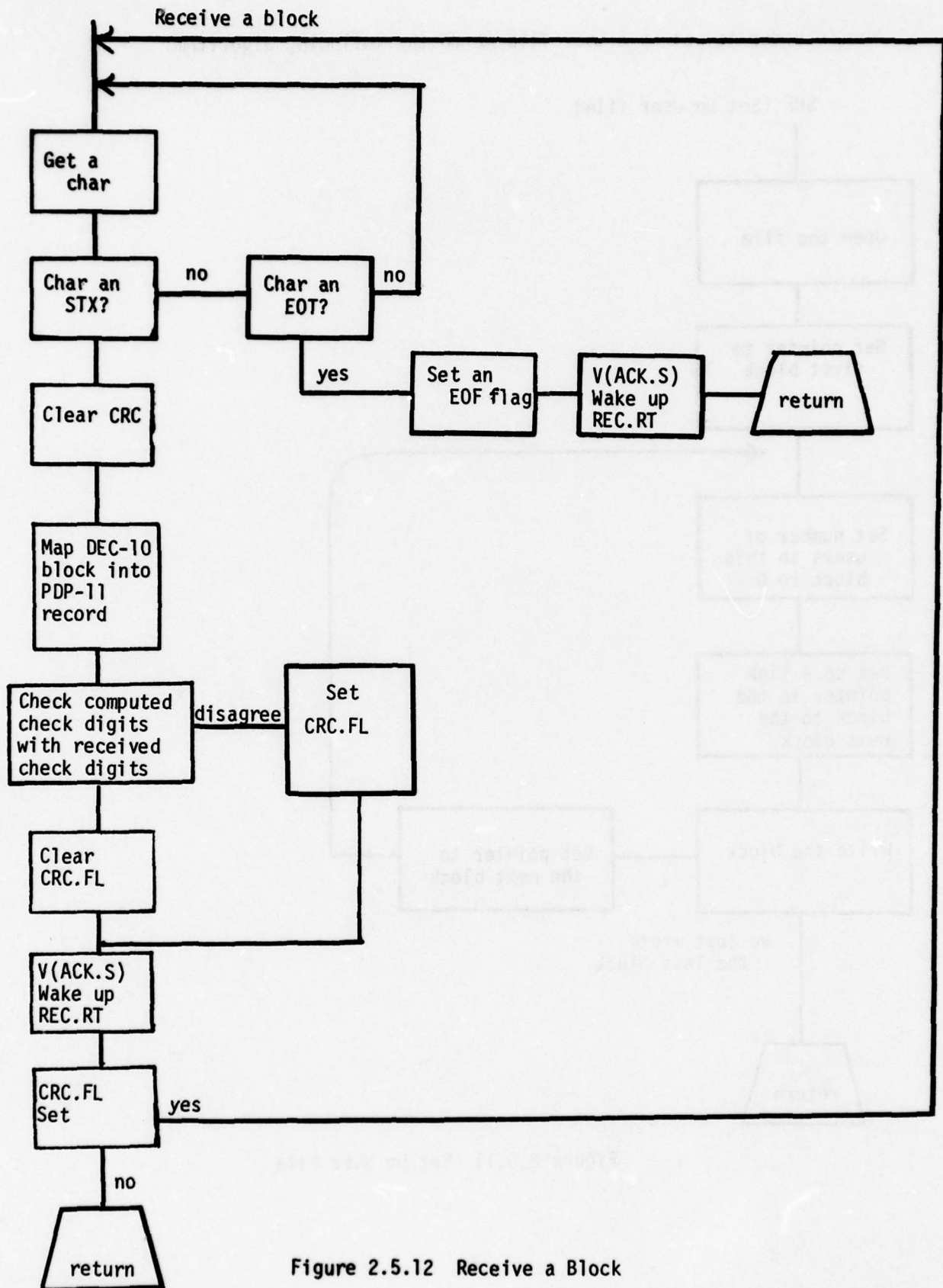


Figure 2.5.12 Receive a Block

The mapping for DEC-10 blocks
to PDP-11/04 records is:

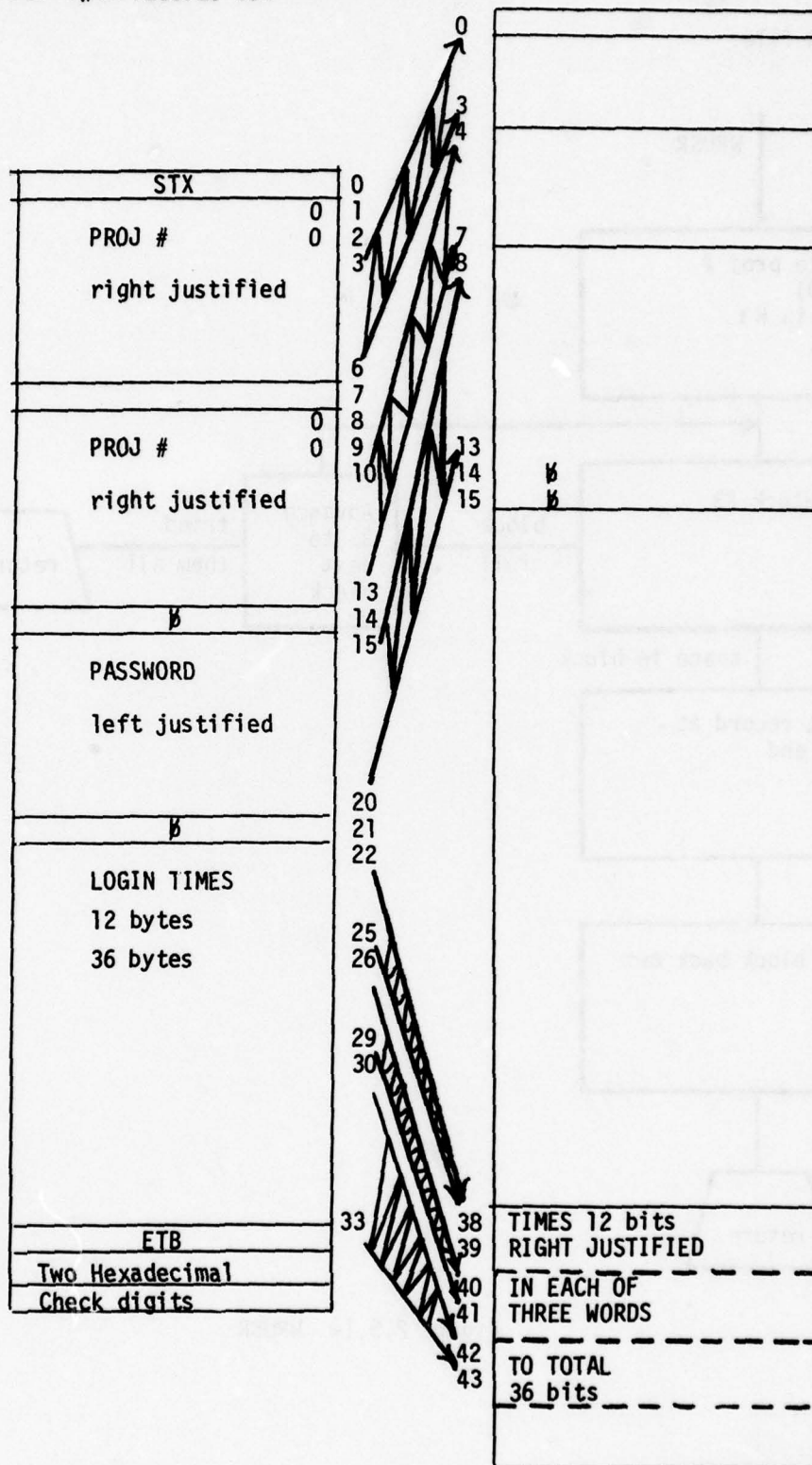


Figure 2.5.13

After receiving a block of data and mapping it to a record we must write it in the USER file.

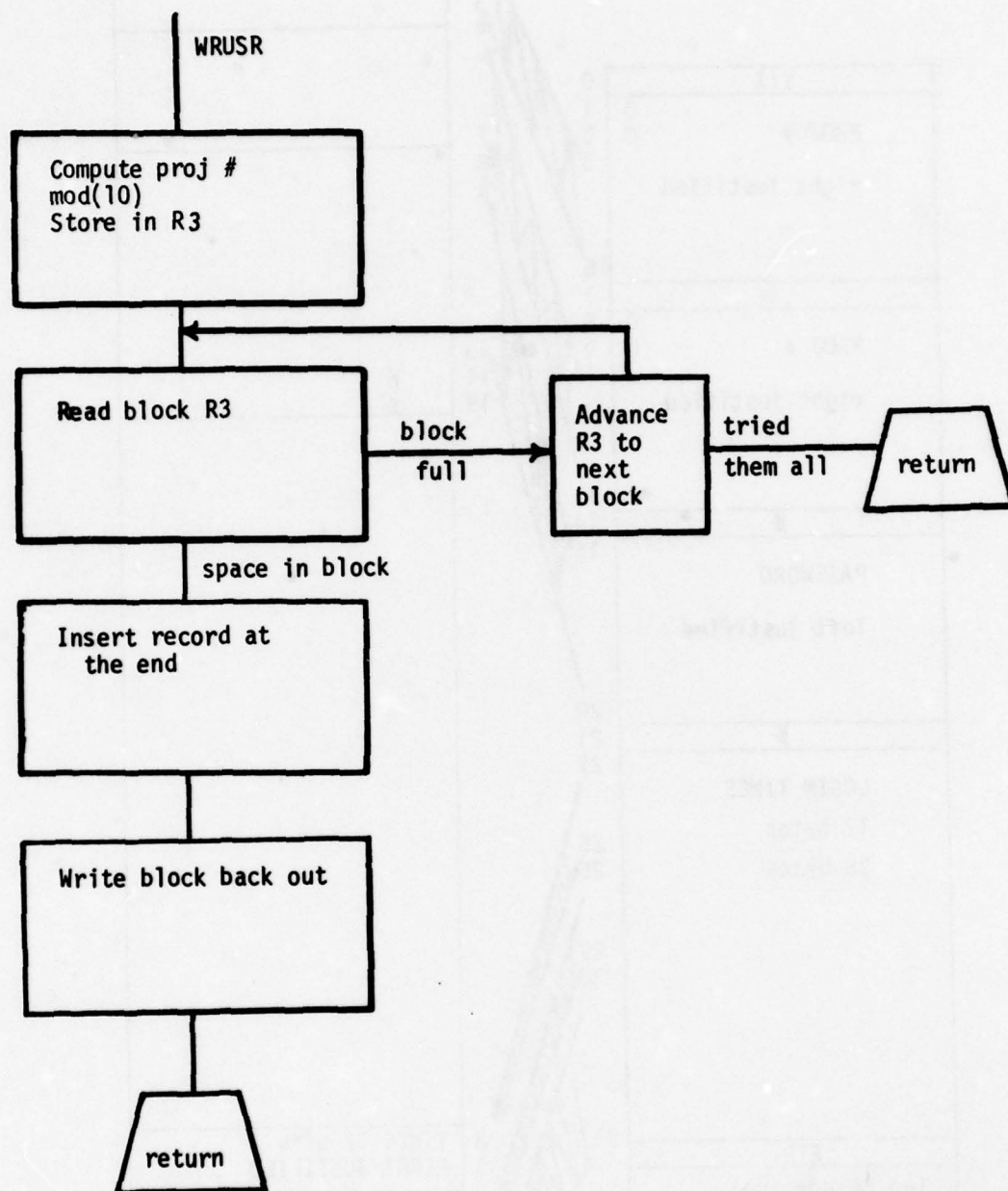


Figure 2.5.14 WRUSR

REC.RT(file ROF)

REC.RT runs under TTIP. It causes TT input to be ignored and uses the line into the DEC 10 to send ACK's and NAK's after each block RECAFIL receives.

It waits on the semaphore ACK.S. When released it checks the flag EOT.FL to see if it should terminate and if not it checks the flag CRC.FL and sends an ACK if it is 0 or NAK if it is 1.

FLOAD

The program FLOAD is written in DEC-10 MACRO assembler. Its introductory documentation is

FLOAD - PROGRAM TO SEND ACCESS AUTHORIZATIONS TO PDP-11'S MACRO #52(610)

FLOAD MAC 2-JAN-78 WHAT THIS PROGRAM DOES

```

65        SUBTTL   WHAT THIS PROGRAM DOES
66
67        :        THIS PROGRAM READS ACCT.EXT (SAME FORMAT AS ACCT.SYS) AND
68        :        SEEDS APPROPRIATE ITEMS OUT OVER AN ASYNCHRONOUS COMMUNICATIONS
69        :        LINE TO A PDP-11, TO BE USED FOR GRANTING OR DENYING ACCESS TO
70        :        THE AFAL PDP-11'S.
71        :
72        :        THIS PROGRAM WORKS WITH VERSION 4 OF ACCT.EXT. FILE FORMAT IS
73        :        ONE HEADER WORD, WITH THE LEFT HALF= TO THE CURRENT VERSION, AND
74        :        THE RIGHT HALF EQUAL TO THE ENTRY SIZE (16OCTAL FOR VERSION 4).
75        :        THE HEADER WORD IS FOLLOWED BY ANY NUMBER OF ENTRIES, EACH OF
76        :        THE LENGTH GIVEN IN THE HEADER. INFORMATION IN EACH ENTRY IS
77        :        POSITIONAL. FOR EXAMPLE, WORD 0 IS ALWAYS THE PPN, WORD 1 IS
78        :        ALWAYS THE PASSWORD, AND SO ON.
79        :
80        :        DATA CURRENTLY SENT IS
81        :
82        :        NAME        WHERE        DESCRIPTION
83        :        ----        -
84        :
85        :        PPN        0        PROJECT-PROGRAMMER NUMBER
86        :        CODE        1        0 TO 6 CHARACTER PASSWORD
87        :        LITIME    5        TIMES AT WHICH THE USER IS ALLOWED TO LOGIN
88        SUBTTL   OUTPUT   FORMAT
89
90        COMMENT
91
92        (SEE FACTS.MAC FOR MORE DETAILED EXPLANATION)
93
94        1.        DEC-10 SENDS <ENQ> TO SIGNAL START OF TRANSMISSION
95
96        2.        DEC-10 SENDS 0 OR MORE BUFFERS OF DATA, IN THE FORMAT GIVEN
97        :        BELOW.    EACH BUFFER IS PRECEDED WITH <STX>
98
99        3.        WHEN THERE ARE NO MORE BUFFERS TO BE SENT, DEC-10 SENDS <EOT>.
100
101
102        BUFFER FORMATS
103
104        <STX>    SIGNAL START OF TEXT FOR ONE RECORD
105
106        ...ALL OF THE DATA
107
108        <ETB>    SIGNALS END OF BUFFER TEXT
109
110        XX        TWO CHECK DIGITS. THESE DIGITS ARE SUCH THAT WHEN THE SUM OF
111        :        ALL CHARACTERS FOLLOWING (NOT INCLUDING) <STX> UP THROUGH
112        :        (AND INCLUDING) <ETH> IS ADDED TO THE VALUE OF THESE TWO DIGITS
113        :        (THE DIGITS ARE HEXADECIMAL) THE LOW ORDER 7 BITS OF THE SUM
114        :        IS ZERO.

```


FLOAD - PROGRAM TO SEND ACCESS AUTHORIZATIONS TO PDP-11'S MACRO #52(610)
 FLOAD MAC 2-JAN-78 12:38 OUTPUT FORMAT

115 SUBTTL OUTPUT FORMAT
 116 CRLF CARRIAGE RETURN LINE FEED TO TERMINATE THE RECORD (ON OUTPUT
 117 FROM DEC-10. ON INPUT, CARRIAGE RETURN IS INPUT, DEC-10.
 118 GENERATES THE LINE FEED).

119 NOTE: BOTH COMPUTERS SHOULD IGNORE NULLS AND RUBOUTS (ASCII CODES
 120 000 AND 177). THE DEC-10 WILL GENERATE EVEN PARITY, BUT WILL
 121 IGNORE PARITY ON INPUT. EACH COMPUTER MAY CHOOSE TO IGNORE
 122 CARRIAGE RETURNS AND LINE FEEDS, ALSO.

125
 126 BUFFER DATA LAYOUT:

128 COL	129 WHAT	130 DESCRIPTION
131 ---	132 ----	133 -----
134 1-6	135 DDDDDD	136 OCTAL PROJECT NUMBER
137 7	138 ,	139 COMMA SEPARATOR
140 8-13	141 DDDDDD	142 OCTAL PROGRAMMER NUMBER
143 14	144 B	145 SPACE SEPARATOR
146 15-20	147 AAAAAA	148 PASSWORD
149 21	150 B	151 SPACE SEPARATOR
152 22-33	153 DDDDDDDDDDDD	154 OCTAL. 12 DIGIT OCTAL NUMBER REPRESENTING 155 36-BIT ALLOWABLE LOGIN TIMES WORD. ON 16-BIT 156 COMPUTERS, IT MAY BE CONVENIENT TO THINK OF THIS 157 AS THREE 4-DIGIT OCTAL NUMBERS, AS FOLLOWS:
158 22-25	159 DDDD	160 OCTAL. FIRST 12 BITS OF LOGIN TIMES WORD (SEE 161 DESCRIPTION OF SNDBLK CALLING SEQUENCE).
162 26-29	163 DDDD	164 OCTAL. SECOND 12 BITS OF LOGIN TIMES WORD
165 30-33	166 DDDD	167 OCTAL. THIRD 12 BITS OF LOGIN TIMES WORD

168
 169
 170
 171
 172
 173
 174
 175
 176
 177
 178
 179
 180
 181
 182
 183
 184
 185
 186
 187
 188
 189
 190
 191
 192
 193
 194
 195
 196
 197
 198
 199
 200

B. Send an Accounting Entries File to the DEC-10.

1. The Sendafile (file SOF) algorithm is as follows

- (1) Send an <ENQ> to the DEC-10
- (2) Wait for an <ACK>
- (3) Open the ENTRIES file
- (4) Read the first block which just contains the number of blocks in the file
- (5) Repeat PROCESS BLOCK until all are processed
- (6) Close the ENTRIES file
- (7) Send an <EOT> to DEC-10 to terminate FACTS
- (8) Put an <EOT> on DZIQ to terminate ACK.RT
- (9) Return

PROCESS BLOCK

- (1) Read a block
- (2) Get the record count
- (3) Repeat STRAN until all are processed
- (4) Return

ENTRYS Block has room for 17 records

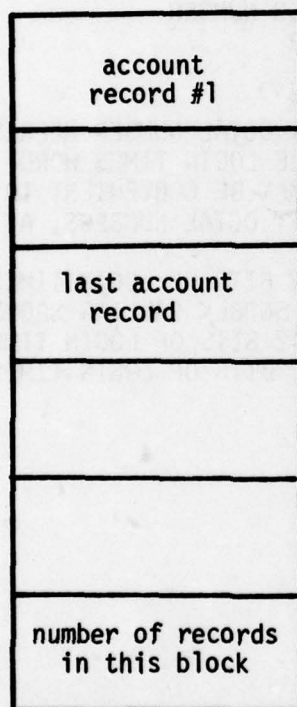
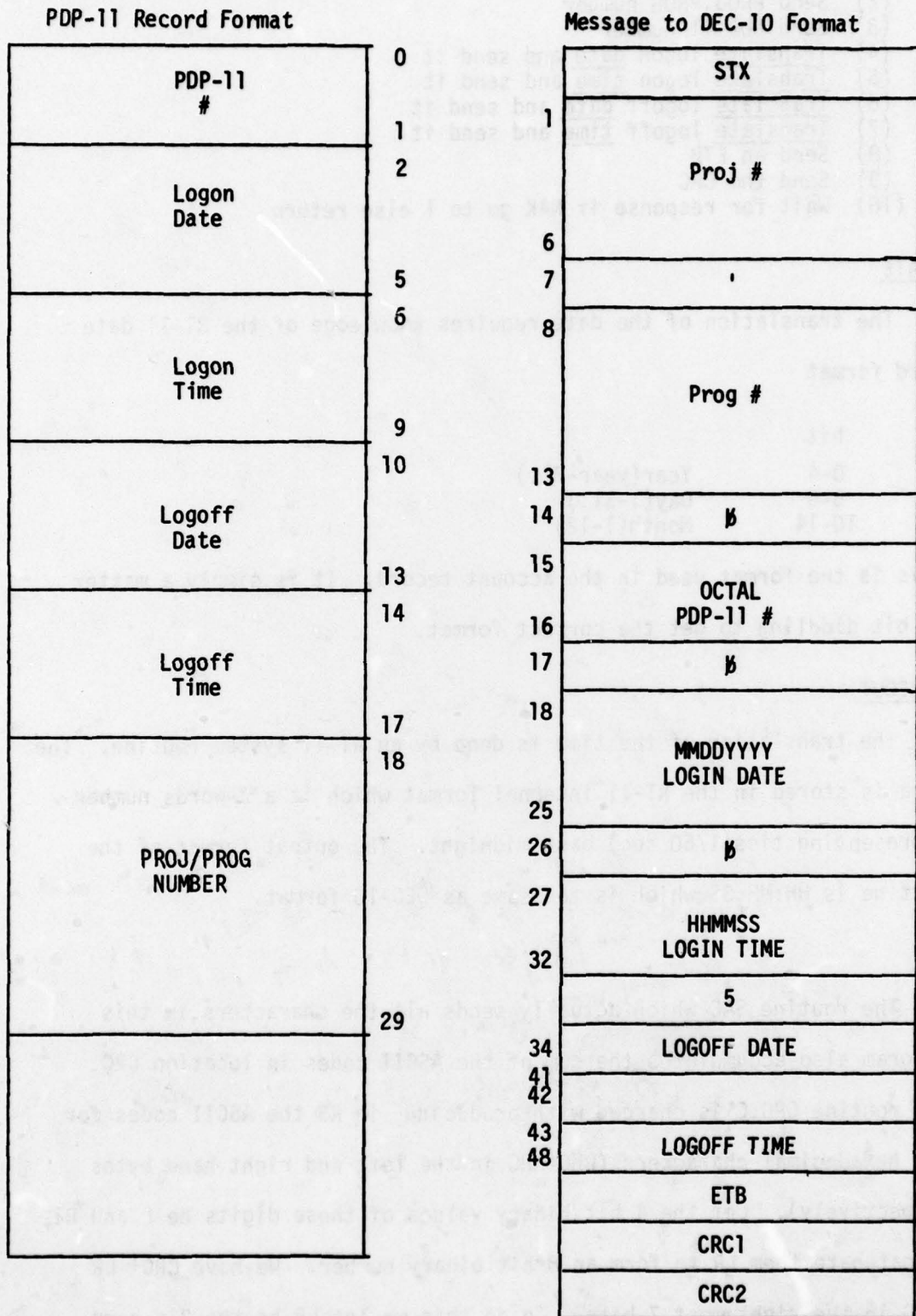


Figure 2.5.15 Process Block

Figure 2.5.16 STRAN

STRAN (Translate and send a record)



STRAN(Translate and send a record)

- (1) Send STX
- (2) Send PROJ, PROG number
- (3) Send PDP-11 number
- (4) Translate logon date and send it
- (5) Translate logon time and send it
- (6) Translate logoff date and send it
- (7) Translate logoff time and send it
- (8) Send an ETB
- (9) Send the CRC
- (10) Wait for response if NAK go to 1 else return

FDATE

The translation of the date requires knowledge of the RT-11 date word format

bit	
0-4	Year(year-72.)
5-9	Day(1-31.)
10-14	Month(1-12)

This is the format used in the account record. It is simply a matter of bit diddling to get the correct format.

TIMEOUT

The translation of the time is done by an RT-11 system routine. The data is stored in the RT-11 internal format which is a 2-words number representing tics(1/60 sec) past midnight. The output format of the routine is HH:MM:SS which is the same as DEC-10 format.

CRC

The routine SAC which actually sends all the characters in this program also accumulates the sum of the ASCII codes in location CRC. The routine CRC.C is charged with producing in R3 the ASCII codes for two hexadecimal characters (LHC-RHC in the left and right hand bytes respectively). Let the 4 bit binary values of these digits be L and R. Concatenate them LR to form an 8 bit binary number. We have CRC+ LR is 0 in the right most 7 bits. To do this we let LR be the 2's comp-

lement of CRC. We then send those characters in the order LHC, RHC.

ACK.RT

ACK.RT runs under the process DZIP and listens for acknowledgments for the program SENDAFILE. It ignores all but 3 characters <ACK>, <NAK>, <EOT>. <ACK>, <NAK> are responses from the DEC-10. When one of these is received a flag is set as to which one it is and SENDAFILE is unblocked. The <EOT> is a signal from SENDAFILE that ACK.RT should terminate.

FACTS

FACTS is a program written in DEC-10 macro the following is the comment section from the program.

FACTS INPUT PDP-11 USAGE INFORMATION MACRO #53(1020)
FACTS MAC 30-Jun-78 14:35 INPUT FORMAT

68 SUBTTL INPUT FORMAT
69
70 COMMENT
71
72 THE PDP-11 SENDS INPUT RECORDS TO THIS PROGRAM, TO BE CONVERTED TO
73 STANDARD DEC-10 FORMAT FOR FACT FILE USAGE ACCOUNTING, AND LOGGED FOR
74 LATER PROCESSING BY SOME OTHER PROGRAM. THE INPUT FORMAT IS AS
75 FOLLOWS:
76
77 A MINI-PROTOCOL HAS BEEN DEFINED FOR THE PDP-11/DEC-10 COMMUNICATIONS
78 NECESSARY TO IMPLEMENT THIS PROGRAM. THIS PROTOCOL UTILIZES CERTAIN
79 CHARACTER SEQUENCES (DEFINED ELSEWHERE AS C.???) TO SIGNAL END OF
80 BUFFER, NEGATIVE ACKNOWLEDGE, ETC. THESE CHARACTERS WILL BE
81 REPRESENTED ON THIS PAGE AS <CHAR> (AND APPEAR IN THE SOURCE AS C.CHAR).
82
83 NOTE: BOTH COMPUTERS SHOULD IGNORE NULLS, RUBOUTS, CARRIAGE RETURNS,
84 AND LINE FEEDS.
85
86 <REQ> SENT BY DEC-10 TO REQUEST ENQ .
87 <ENQ> SENT BY PDP-11 TO INITIATE PROCESSING
88 <ACK> POSITIVE ACKNOWLEDGE OF LAST MESSAGE
89 <NAK> NEGATIVE ACKNOWLEDGE OF LAST MESSAGE
90 <STX> START OF TEXT (ONE INPUT RECORD FOLLOWS)
91 <EOT> END OF TRANSMISSION (PDP-11 IS ALL DONE)
92 <ETB> END OF TEXT (FOLLOWS EACH INPUT RECORD).
93
94 THESE CHARACTERS DO NOT HAVE THEIR STANDARD ASCII DEFINITIONS, BUT ARE
95 ALPHABETIC CHARACTERS.
96
97 FOLLOWING IS A DESCRIPTION OF A TYPICAL INTERACTION WITH THE PDP-11:
98
99 L0: DEC-10 SENDS <REQ>.
100
101 L1: PDP-11 SENDS <ENQ>.
102
103 L2: DEC-10 IS WAITING FOR <ENQ>, AND DISCARDS ALL OTHER CHARACTERS
104 UNTIL <ENQ> IS SEEN. WHEN DEC-10 RECEIVES ENQ , IT RESPONDS
105 WITH <ACK>.
106
107 L3: PDP-11 SENDS EITHER <STX> (FOLLOWED BY RECORD) OR <EOT>
108 L4:
109 IF DEC-10 RECEIVED <STX> GOTO L4
110 IF DEC-10 RECEIVED <EOT> GOTO L5
111 ELSE, FATAL ERROR.
112
113 L5: (DEC-10 RECEIVED <STX>).
114 WHILE (NEXT CHARACTER .NE. <ETB>) DEC-10 READS CHARACTERS
115 AND STORES THEM IN A BUFFER.


```

116      L5:  INPUT  FORMAT
117      <ETB> TERMINATES BUFFER, AND IS FOLLOWED BY TWO CHECK DIGITS.
118      THESE CHECK DIGITS ARE HEXADECIMAL, AND ARE SUCH THAT THE SUM
119      OF ALL CHARACTERS IN THE RECORD (EXCLUSIVE OF <STX>, BUT
120      INCLUDING <ETB> AND OF THE VALUE OF THE TWO DIGIT HEX NUMBER
121      IS ZERO IN ITS LOW ORDER 7 BITS.
122      THE CHECK DIGITS ARE FOLLOWED BY A CARRIAGE RETURN (THE
123      DEC-10 MONITOR APPENDS A LINE FEED, SO THIS PROGRAM SEES
124      THE TWO CHARACTERS).
125
126      IF CHECKSUM IS CORRECT, DEC-10 SENDS <NAK>, PDP-11 WILL
127      RETRANSMIT THAT RECORD. IF CORRECT, DEC-10 SENDS <ACK>.
128      GOTO L3.
129
130      L6:  (DEC-10 RECEIVED <EOT>).
131      DONE.  CLOSE FILES AND EXIT
132
133      .....
134
135      RECORD FORMAT:
136
137      COL      FORMAT      DESCRIPTION
138      ---      -
139      1-6      DDDDDD      OCTAL PROJECT NUMBER
140      7        ,          COMMA SEPARATION
141      8-13     DDDDDD      OCTAL PROGRAMMER NUMBER
142      14       B          SPACE SEPARATOR
143      15-16    DD         OCTAL PDP-11 NUMBER
144      17       B          SPACE SEPARATOR
145      18-25    MMDDXXXX    LOGIN DATE
146      26       B          SPACE SEPARATOR
147      27-32    HHMMSS     LOGIN TIME
148      33       B          SPACE SEPARATOR
149      34-41    MMDDXXXX    LOGOFF DATE
150      42       B          SPACE SEPARATOR
151      43-48    HHMMSS     LOGOFF TIME
152
153      SAMPLE:
154      123456,132542 02 12311977 122344 12311977 130252
155
156      (USER (123456,132542) LOGGED ON PDP-11#02 ON 31-DEC-77 AT 12:23:44,
157      AND LOGGED OFF 31-DEC-77 AT 13:02:52).
158
159
160      :END OF COMMENT

```

C. Transfer a file to the DEC-10 for development purposes.

In order to transfer a file to the DEC-10

- (1) The file must be named DX1:DEC10.LST
- (2) The DEC-10 must be prepared to receive the file. The easiest way to do this is to call the DEC-10 version program of PIP. Set up a transfer from TT: to any file you wish
- (3) Type a +C followed by a +R. You will see the file echoed on the terminal as it is transferred
- (4) When the file is transferred, hit a CR followed by a +Z which will close the file

1. TRANF (SOF)

- (1) Open the file
- (2) Repeat PROCESS BLOCK until end of file
- (3) Close the file

PROCESS BLOCK

- (1) Read block
- (2) Note if end of file
- (3) Repeat Send Char until all sent

SEND CHAR

- (1) Get a character buffer and put a character in it
- (2) Wait until DEC 10 wants a character(flag SEND.FL)
Comment: Flow control is managed by DZIP which is monitoring the +S's, +Q's sent by the DEC-10 for this purpose
- (3) Wait until the output queue DZIQ is not full
- (4) Put the character in the queue

2.6 Communications Between MIU and PDP-11/04

The communications between each MIU and the PDP-11/04 is accomplished via an asynchronous hardwire RS-232 communications link with a null modem.

The characteristics of the asynchronous characters are

- a) 4800 baud, full duplex
- b) 8 bits of data per character
- c) odd character parity
- d) two stop bits
- e) ASCII Code

The flow of information between the PDP-11/04 and the Microprocessor Interface Unit is controlled by the Microprocessor Interface Unit. It initiates all conversations. It waits between 1 and 2 seconds for a response from the PDP-11/04. If an acceptable response is not received within the time interval, it retries the transmission twice. If the communication of a message is not completed it enters a default mode which logs the user "on" or "off" as he desires.

All messages are 32 (decimal) bytes long, padded as necessary with garbage. All messages are preceded by the character "@" which serves to clear the message buffer. The message packets are described in Table 2.6

PDP-11 To/From Intel 8080 Message Packets

All messages are 32 (decimal) bytes long, padded as necessary with garbage.

LOGON

8080 to 11	bytes	1-4	project number (octal ascii) - right justified filled with leading zeros
	bytes	5-8	programmer number (octal ascii) - same
	bytes	9-16	user password (ascii) - left justified, filled with trailing blanks.
	bytes	17-32	not used

11 to 8080	If valid logon:		
	byte	1	letter "V"
	bytes	2-10	date, "dd-mm-yy"
	bytes	11-18	time, "hh:mm:ss"
	bytes	19-32	not used

	If non-valid logon:		
	byte	1	not letter "V"
	byte	2	error code-
			"N" invalid ppn
			"P" incorrect password
			"T" not allowed at this time of day
	bytes	3-32	not used

LOGOFF

8080 to 11	byte	1	letter "L"
	bytes	2-32	not used
11 to 8080	byte	1	letter "L"
	bytes	2-10	date, "dd-mm-yy"
	bytes	11-17	time, "hh:mm:ss"
	bytes	18-26	elapsed time, "hh:mm:ss"
	bytes	27-32	not used

Table 2.6

3.0 Conclusions and Recommendations

a. The monitoring system installed in the PDP-11's in the Avionics Laboratory performs as designed. It collects accounting information and controls access for eight PDP-11's. It could easily be expanded to fifteen PDP-11's. If additional units are to be constructed, I would propose the system be redesigned using a printed circuit board. If more than 15 units are to be monitored, additional hardware, another DZ-11, and software modifications would be necessary in the PDP-11/04.

b. There is a lot of user hostility toward the system. The current habits of many of the users should be studied to determine the source of the hostility. It may be that a modified version or different system is necessary to solve this problem.

c. The monitoring system was designed for single user PDP-11 systems. It does not provide effective accounting or control on systems operated by multiple users in a time-sharing mode using RSX-11 such as the PDP-11/50 system in CSEL.

d. On several occasions, it has been reported that a user would be logged onto the system and then immediately logged off. This might occur if the bootstrap loader was attempting to load from the disc and was detecting errors and made multiple retries. The problem was not repeatable by maintenance personnel. I suspect marginal reading of a disc pack caused by alignment problems. If this should become a serious problem, the mode of logging off by multiple INIT's can be disabled by connecting pin 2 on IC18 of the special logic board to ground. Log off could still be accomplished by powering down the PDP-11 or by a third mode not user tested but contained in the hardware.

e. A third mode of logging off of the PDP-11 exists in the hardware. We had some problems in early marriage tests with the interface which made this third mode appear to be incompatible with some of the PDP-11's. I

suspect that those problems are not problems with the procedure and merit additional study.

The third method involves using the switches on the front panel of the CPU. The log off is initiated by any data out operation to the PDP-11 address specified by the M105 card contained in the microprocessor interface unit. The "Select 0" address on the M105 card is used to set the microprocessor to a state in which it is waiting for an INIT signal on the UNIBUS. When the INIT signal occurs the microprocessor interface requests the bus and enters the log off sequence.

Third Method for Logging Off

1. Load address on M105 card into switch register.
2. Deposit to that address any data.
3. Load address of boot to execute a system boot.
4. Press start switch to start execution.
5. Interface will take control of the bus execute the log off sequence and wait for next user to log on.

f. Currently the software does not permit the Data Logging System, PDP-11/04, to be operated simultaneously collecting accounting data, ACCT, and communicate with the DEC-10. This feature could be added, but it could require a major rewriting of the software.

g. The system currently runs under DEC's RT-11 version 1 operating system. To gain some enhanced features of later versions of the operating system would require some software modifications. We do not have copies of the documentation describing later versions of RT-11, so it is not possible for us to estimate the scope of such changes.